UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE INFORMÁTICA

TESIS DOCTORAL

Knowledge acquisition for autonomic network management in emerging self-organizing architectures

Adquisición de conocimiento para la gestión autónoma de redes en arquitecturas auto-organizadas emergentes

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

Marco Antonio Sotelo Monge

Director

Luis Javier García Villalba

Madrid
Ed. electrónica 2019

Knowledge Acquisition for Autonomic Network Management in Emerging Self-Organizing Architectures

Adquisición de Conocimiento para la Gestión Autónoma de Redes en Arquitecturas Auto-Organizadas Emergentes

Thesis by

Marco Antonio Sotelo Monge

In Partial Fulfillment of the Requirements for the Degree of
Doctor por la Universidad Complutense de Madrid en el
Programa de Doctorado en Ingeniería Informática

Advisor

Luis Javier García Villalba

Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid

Madrid, 2018

Knowledge Acquisition for Autonomic Network Management in Emerging Self-Organizing Architectures

Thesis by

Marco Antonio Sotelo Monge

In Partial Fulfillment of the Requirements for the Degree of
Doctor por la Universidad Complutense de Madrid en el
Programa de Doctorado en Ingeniería Informática

Advisor

Luis Javier García Villalba

Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid

Madrid, 2018

Adquisición de Conocimiento para la Gestión Autónoma de Redes en Arquitecturas Auto-Organizadas Emergentes

TESIS DOCTORAL

Memoria presentada para obtener el título de
Doctor por la Universidad Complutense de Madrid
en el Programa de Doctorado en Ingeniería Informática

Marco Antonio Sotelo Monge

Director

Luis Javier García Villalba

Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid

Madrid, 2018

Dissertation submitted by Marco Antonio Sotelo Monge to the *Facultad de Informática* of the *Universidad Complutense de Madrid* in Partial Fulfillment of the Requirements for the Degree of *Doctor por la Universidad Complutense de Madrid en el Programa de Doctorado en Ingeniería Informática*.

Title:

Knowledge Acquisition for Autonomic Network Management in Emerging Self-Organizing Architectures

PhD Student:

Marco Antonio Sotelo Monge (masotelo@ucm.es)
Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid
28040 Madrid, Spain

Advisor:

Luis Javier García Villalba (javiergv@fdi.ucm.es)

This work has been done within the Group of Analysis, Security and Systems (GASS, `http://gass.ucm.es/`), Research Group 910623 from the Universidad Complutense de Madrid (UCM) as part of the activities of the research project funded by the European Commission Horizon 2020 Programme under Grant Agreement number H2020-ICT-2014-2/671672-SELFNET (Framework for Self-Organized Network Management in Virtualized and Software Defined Networks).

Tesis Doctoral presentada por el doctorando Marco Antonio Sotelo Monge en la Facultad de Informática de la Universidad Complutense de Madrid para la obtención del título de Doctor por la Universidad Complutense de Madrid en el Programa de Doctorado en Ingeniería Informática.

Título:

Adquisición de Conocimiento para la Gestión Autónoma de Redes en Arquitecturas Auto-Organizadas Emergentes

Doctorando:

Marco Antonio Sotelo Monge (masotelo@ucm.es)
Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid
28040 Madrid, España

Director:

Luis Javier García Villalba (javiergv@fdi.ucm.es)

Esta tesis doctoral ha sido realizada dentro del grupo de investigación GASS (Grupo de Análisis, Seguridad y Sistemas, grupo 910623 del catálogo de grupos reconocidos por la UCM) como parte de las actividades del proyecto de investigación SELFNET (Framework for Self-Organized Network Management in Virtualized and Software Defined Networks) financiado por la Comisión Europea dentro del Programa Marco de Investigación e Innovación Horizonte 2020 (H2020-ICT-2014-2/671672-SELFNET).

This thesis is dedicated to
my mother and to my siblings

La presente tesis está dedicada
a mi madre y a mis hermanos

Acknowledgments

I want to thank my advisor Luis Javier García Villalba for his support throughout these years.

Thanks to my family, from whom I have always got the best.

Special thanks to my friend Jorge Maestre Vidal. His guidance and dedication to this thesis were priceless.

Thanks also to the Programa Nacional de Becas y Crédito Educativo (PRONABEC) of the Peruvian Ministry of Education.

This work has been done within the Group of Analysis, Security and Systems (GASS, `http://gass.ucm.es/`), Research Group 910623 from the Universidad Complutense de Madrid (UCM) as part of the activities of the research project funded by the European Commission Horizon 2020 Programme under Grant Agreement number H2020-ICT-2014-2/671672-SELFNET (Framework for Self-Organized Network Management in Virtualized and Software Defined Networks).

Agradecimientos

Quiero agradecer a mi asesor Luis Javier García Villalba por su apoyo durante estos años.

Gracias a mi familia, de quienes siempre he obtenido lo mejor.

Un agradecimiento especial a mi amigo Jorge Maestre Vidal. Su guía y dedicación a esta tesis tienen un valor incalculable.

Gracias también al Programa Nacional de Becas y Crédito Educativo (PRONABEC) del Ministerio de Educación del Perú.

Esta tesis doctoral ha sido realizada dentro del grupo de investigación GASS (Grupo de Análisis, Seguridad y Sistemas, grupo 910623 del catálogo de grupos reconocidos por la UCM) como parte de las actividades del proyecto de investigación SELFNET (Framework for Self-Organized Network Management in Virtualized and Software Defined Networks) financiado por la Comisión Europea dentro del Programa Marco de Investigación e Innovación Horizonte 2020 (H2020-ICT-2014-2/671672-SELFNET).

Contents

List of Figures

List of Tables

List of Acronyms

3GPP	*Third Generation Partnership Project*
5G	*Fifth Generation Mobile Network*
5GMF	*Fifth Generation Mobile Communications Promotion Forum*
ADB	*Aggregated Data Bundle*
AMQP	*Advanced Message Queuing Protocol*
ANN	*Artificial Neural Network*
API	*Application Programming Interface*
BN	*Bayesian Network*
BSS	*Business Support System*
CAPEX	*Capital Expenditures*
D2D	*Device to Device Communication*
DDoS	*Distributed Denial of Service*
DNS	*Domain Name System*
DPI	*Deep Packet Inspection*
EDoS	*Economic Denial of Sustainability*
ENISA	*European Union Agency for Network and Information Security*
EPC	*Evolved Packet Core*
ETSI	*European Telecommunications Standards Institute*
HoN	*Health of Network*
IaaS	*Infrastructure as a Service*
IDS	*Intrusion Detection System*
IETF	*Internet Engineering Task Force*

IoT	*Internet of Thing*
ISRA	*Information Security Risk Assessment*
KPI	*Key Performance Indicator*
LTE	*Long Term Evolution*
M2M	*Machine to Machine Communication*
MIMO	*Multiple Input Multiple Output*
MME	*Mobile Management Entity*
NaaS	*Network as a Service*
NAT	*Network Address Translation*
NF	*Network Function*
NS	*Network Service*
NFV	*Network Function Virtualization*
NFVI	*Network Function Virtualization Infrastructure*
NFVO	*Network Function Virtualization Orchestrator*
NOS	*Network Operating System*
NSSA	*Network Security Situational Awareness*
ODL	*OpenDaylight*
ONF	*Open Networking Foundation*
OPEX	*Operational Expenditures*
OSS	*Operational Support System*
PaaS	*Platform as a Service*
QoE	*Quality of Experience*
QoS	*Quality of Service*
RAN	*Radio Access Network*
RFC	*Request for Comment*
SA	*Situational Awareness*
SDN	*Software Defined Networking*
SELFNET	*Self-Organized Network Management in Virtualized and Software Defined Networks*

SLA	*Service Level Agreement*
SON	*Self Organizing Networks*
SVM	*Support Vector Machine*
VIM	*Virtual Infrastructure Manager*
VNF	*Virtual Network Function*
VPN	*Virtual Private Network*

Abstract

Emerging network scenarios are characterized by intensive access to a wide range of services and applications that have increased the demands of communication networks. The traditional network management models have been characterized by a high dependence on the human factor to carry out network configuration and maintenance tasks. This situation has become less sustainable in mobile networks not only due to the associated operational (COPEX) and capital investment costs (CAPEX), but also due to the complexity they have acquired when facing the exponential immersion of mobile devices. These aspects have led to the emergence of the fifth generation of mobile networks, characterized by ambitious performance indicators that must be fulfilled to meet the agreed service levels.

5G networks are grounded on the integration of technological advances promoted by Software Defined Networks (SDN), Network Functions Virtualization (NFV), artificial intelligence, among others, that have shifted the traditional network management paradigms towards a self-organized and software-driven approach. It is therefore essential to develop analytical methods based on artificial intelligence techniques to obtain knowledge about the state of the network in order to infer possible situations that might put the operativity of the network at risk. To this end, this research focuses on the study of knowledge acquisition methods aimed to introduce self-organization capabilities in 5G networks. It should be also noted that this thesis has been framed in the SELFNET project (Framework for Self-organized Network Management in Virtualized and Software Defined Networks) funded by the Horizon 2020 program of the European Commission, whose purpose is the design and implementation of an autonomic management framework for 5G networks. As a result, SELFNET has provided a reference architecture for the development of the proposals conducted as part of this research.

This thesis also provides a review of the state of the art in 5G networks and their supportive technologies. Likewise, diverse prediction and pattern recognition methods have been studied in detail in order to conduct analytical processes focused on the acquisition of knowledge about the monitored networks.

The performed research proposed also an analytical framework oriented to the acquisition of knowledge in emerging network scenarios based on the architectural principles described above. The evaluation of such proposal has shown its effectiveness in view of the results obtained in the experimentation stage. This evaluation has been conducted both at individual component level and at use case level.

On the other hand, detection methods for two major threats have been studied from the perspective of emerging communication scenarios. The first of them focuses on

the detection of Economic Denial of Sustainability (EDoS) attacks in cloud computing environments. This approach has been based on the study of anomalous behaviors of the entropy degree measured on application-level metrics. The effectiveness of the proposal has led to its study in self-organized network scenarios, where it has been also demonstrated acceptable levels of precision in the light of the results. The second detection method has focused on inferring the participation of a network device in a Distributed Denial of Service (DDoS) attack. The proposed model has been based on the knowledge acquisition principles established as part of this research, demonstrating also acceptable levels of accuracy in the evaluated experimental scenarios.

Finally, the conducted research has opened interesting lines of future work described at the end of this thesis.

Keywords: 5G, cloud computing, DDoS, EDoS, prediction, pattern recognition, reasoning, Self-Organizing Networks, Software Defined Networking.

Resumen

Los escenarios de red emergentes están caracterizados por el acceso intensivo a una amplia gama de servicios y aplicaciones que han incrementado las exigencias de las redes de comunicación. Los modelos de gestión de red tradicionales se han caracterizado a su vez por una alta dependencia del factor humano para llevar a cabo tareas de configuración y mantenimiento de la red. Esta situación se ha hecho menos sostenible en las redes móviles no sólo por los costes operacionales y de inversión de capital asociados, sino también por la complejidad que estas han adquirido ante la inmersión exponencial de dispositivos móviles. Tales aspectos han motivado el surgimiento de la quinta generación de redes móviles, caracterizadas por indicadores de desempeño ambiciosos que deben cumplirse para satisfacer los niveles de servicio acordados.

Las redes 5G se sustentan en la integración de los avances tecnológicos promovidos por las redes definidas por software (SDN), la virtualización de funciones de red (NFV), la inteligencia artificial, entre otras, que han supuesto un cambio en los paradigmas tradicionales de gestión hacia un enfoque autoorganizado y dirigido por software. Es imprescindible por lo tanto el desarrollo de métodos de análisis basados en inteligencia artificial para la obtención de conocimiento acerca del estado de la red con el fin de inferir posibles situaciones que puedan poner en riesgo la operatividad de la red. Con este propósito, la presente investigación se enfoca en el estudio de métodos de adquisición de conocimiento orientados a introducir capacidades de auto organización en redes 5G. Debe tenerse en cuenta también que esta tesis se ha enmarcado en el proyecto SELFNET (Framework for Self-organized Network Management in Virtualized and Software Defined Networks) perteneciente al programa Horizonte 2020 de la Comisión Europea, cuyo propósito es el diseño e implementación de un marco de gestión autónoma para redes 5G. Como resultado, SELFNET ha proporcionado una arquitectura de referencia para el desarrollo de las propuestas que conforman esta investigación.

Esta tesis ofrece además una revisión del estado del arte en redes 5G y las tecnologías en las que se apoya. Asimismo, diversos métodos de predicción y reconocimiento de patrones han sido estudiados en detalle con el fin de conducir procesos analíticos enfocados en la adquisición de conocimiento sobre las redes monitorizadas.

La investigación realizada propone también un marco de análisis orientado a la adquisición de conocimiento en escenarios de red emergentes sustentado en los principios arquitectónicos descritos anteriormente. La evaluación de esta propuesta ha demostrado su efectividad a la vista de los resultados obtenidos en la etapa de experimentación. Esta evaluación se ha realizado tanto a nivel de componentes individuales como a nivel de casos

de uso.

Por otra parte, métodos de detección para dos tipos principales de amenazas han sido estudiados desde la perspectiva de los escenarios de comunicación emergentes. El primero de ellos se enfoca en la detección de ataques de denegación de la sostenibilidad económica (EDoS) en entornos de computación en la nube. Dicha aproximación se ha sustentado en el estudio de comportamientos anómalos del grado de entropía observado en métricas a nivel de aplicación. La efectividad de esta propuesta ha dado lugar a su estudio en escenarios de red auto organizados, demostrando también un alto nivel de precisión sobre la base de los resultados obtenidos. El segundo método de detección, se ha enfocado en inferir la participación de un dispositivo de red en un ataque de denegación de servicio distribuido (DDoS). El modelo propuesto se ha sustentado en los principios de adquisición de conocimiento establecidos como parte de esta investigación, y ha demostrado niveles aceptables de precisión en los escenarios evaluados.

Finalmente, la investigación realizada ha abierto interesantes líneas de trabajo futuro descritas al final de esta tesis.

Palabras clave: 5G, computación en la nube, DDoS, EDoS, predicción, razonamiento, reconocimiento de patrones, redes autoorganizadas, redes definidas por software.

Chapter 1

Introduction

Information technology has brought an unlimited number of opportunities to overcome a broad range of societal challenges with innovative solutions characterized by higher productivity and lower response times. This fact was clearly evidenced throughout the last decades by the emergence of communication networks, which have involved significative changes motivated by a steady immersion of the Internet in almost every aspect of our lives. With the rapid proliferation of mobile devices, this immersion has been strengthened since the higher demand of ubiquitous network services opened new challenges for all technology providers, but particularly for telecommunication operators.

Network architectures have strained to evolve as the operational context has gained complexity. Traditional networks have been limited by the rigidity of both physical devices and telecommunication infrastructures on which standardization processes (e.g. signaling or protocols) were slow, hence the agile delivery of network solutions has been constrained. This situation has been worsened by the tight integration of control and data planes in the network devices, where the limitation of computing and networking resources has raised restrictiveness when dealing with agility. For instance, throughput or memory capacity have been critical in the manufacturing of a network device in order to be compliant with a specific protocol family. Likewise, configuration tasks have also been manually performed, so that tehy have caused higher network downtimes with less operational costs efficiency. However, and despite the intrinsic network limitations, significative technology enhancements have been achieved as network generations have evolved. For instance, in the overall perceived Quality of Experience (QoE) in 2G mobile networks compared with 4G. Such enhancements have been fostered by novel trends such as Software Defined Networking (SDN), Network Function Virtualization (NFV), and other technology enablers; thus laying the foundations for the fifth generation of mobile networks.

5G networks should provide a sustainable and scalable network infrastructure to meet the exponentially-increasing demands on mobile broadband access [5G-18], while leveraging competitiveness, standardization and faster innovation. 5G networks were designed to meet prominent requirements [NGM15] in system performance, enhanced services provision, deployment times, as well as operational, energy and management efficiency. 5G networks should guarantee several outstanding attributes such as faster recovery times, higher traffic demands, higher Quality of Service (QoS) and Quality of

Experience (QoE), lower operational (OPEX) and capital (CAPEX) expenditures, among others. This generational change is driven by economic, societal and operational trends to conform the global network of the 21st century [Eur09].

Network management poses in turn new challenges for 5G networks to be faced not only with automation efficiency but also with the development of smart self-organizing schemas. The inclusion of artificial intelligence is thereby mandatory to shift from a policy-based configuration and incident response model towards a self-management approach grounded on reasoning and machine learning capabilities. This paradigm shift entails the acquisition of knowledge from the managed network in order to timely decide the most suited actions to be enforced when an incident arises. It should be also noted that conducting advanced analytical processes requires computational capacity as well as specialized software elements seamlessly orchestrated to produce real-time responses to a range of network problems. For this reason, the separation of the data, control and management planes plays an essential role since the logic for handling network traffic (e.g. by deploying security measures, adjusting bandwidth consumption, etc.) can be leveraged to high-level software applications. Even though a software-driven management model can open an unlimited number of alternatives, network management in 5G is challenging enough to cope with the requirements of next generation networks. Hence, it also raises new opportunities for the research community.

Given the complexity of emerging 5G network architectures, incident management becomes more critical to fulfill the agreed service levels. Timely responses to restore the normal operational conditions of the network are not straightforward and require a proper understanding of the monitored context for conducting effective decision-making processes. Information technology standards (such as ISO or NIST) and best-practices frameworks (such as COBIT or ITIL) suggest performing incident management processes by means of: the framing of the observed context, the assessment of its potential risks, the monitoring of the identified threats and the deployment of timely response actions. This process stresses the importance of conducting a complex analysis of the monitored data in order to detect or anticipate the occurrence of situations that might disrupt the agreed service levels. Because of the heterogeneity of the monitored scenarios, which will be considerably higher in 5G networks, the enforcement of appropriate countermeasures or preventive responses should rely on effective knowledge acquisition processes adapted to different use cases. Thereby, important research lines are posed towards the accomplishment of this goal.

To contribute with the development of advanced self-organizing capabilities adapted to the autonomic management approach foreseen in 5G networks, the present thesis tackles the study of knowledge-based analysis methods targeted on inferring potential network incidents in emmerging communication scenarios. The conducted research takes advantage of the innovations posed by the supportive technologies of 5G networks when laying out architectural considerations aligned with the design principles of 5G. Hence, granting self-organizing capabilities to accomplish their performance indicators. The conducted research proposed new approaches to tackle with the challenges of network incident management, and their outcomes have suggested promising results while rasising interest research lines for future work.

1.1 Research Problem

This section outlines the main areas taken as the baseline for conducting the present thesis. They pose the research problem and objectives which rule the development of this research.

Current technological trends arise complex requirements to be addressed, and the solutions to cope with them involve a seamless integration of novel concepts applied to emerging scenarios. In the area of modern communication networks, research efforts have been committed to deliver innovative solutions for both common and novel problems related with network management. In particular, the application of data analytics in network management for achieving self-organizing features has drawn the attention of the research community, being the main research topic of this thesis as well .

With the emergence of the fifth generation of mobile networks, some research projects have been undertaken towards the creation of autonomic management models aimed to fulfill the Key Performance Indicators (KPIs) posed by 5G. They share the common concern of dealing with massive amounts of data originated at different architectural levels of the network infrastructure, on which the application of suitable analysis techniques demands a proper comprehension of the operational context. Therefore, the enforcement of preventive or reactive countermeasures to mitigate network incidents and likely service degradation situations depends on the correct modeling of the analyzed domain driven by cognitive approaches. Since 5G technology is currently under development, the implementation of such complex networks intended to support millions of connected devices lay a major challenge in the design of analytical components suited for such architectures. Likewise, those components should play an essential role in the accomplishment of self-organizing capabilities through their seamless interaction with other 5G architectural entities. Both the role of the autonomic management entities and the architectural design considerations pose important fields of study throughout this research.

Furthermore, this thesis has been framed into the SELFNET project. SELFNET (Self-Organized Network Management in Virtualized and Software Defined Networks) is a research project funded by the European Commission under the H2020 programme, which is aimed to develop a management framework for achieving self-organizing capabilities in 5G networks. For this purpose, SELFNET combines leading technologies such as Software Defined Networking (SDN), cloud computing, Network Function Virtualization (NFV), Artificial Intelligence (AI), among others, for the designing and implementation of a multi-tenant aware management framework targeted to address three use cases: self-optimization, self-healing and self-protection. SELFNET considers the vast heterogeneity of the operational domains as well, and defines in turn a multi-layered architecture entailing to distinguish data, control and management planes for the automatic provision of responses to network incidents. This project has provided a reference architecture on which the present research has been grounded. There, the core self-organizing functionalities of SELFNET are conducted by the Autonomic Layer, where analytical approaches should be explored for the accomplishment of efficient network

management. Consequently, the Autonomic Layer has been the component of interest of the present thesis.

The literature review disclosed also that data analysis methods in 5G networks have been barely addressed by the research community, being this the reason why the contributions of this thesis have been oriented towards their study. In this landscape, it has been hypothesized that knowledge-based methods can be suited to conduct data analysis processes aimed on the inference of situations that might disrupt the normal operation of the monitored network. It has led to the distinction of two major research stages: the study of a general-purpose knowledge acquisition approach framed into the 5G architecture, and an in-depth study of well-defined use cases where the effectiveness of the proposed approach is validated.

It has been outlined the importance of having a deep understanding about the network in order to enable effective decision-making processes on the basis of the comprehension of the current network status and the projected scenarios. In that direction, the Situational Awareness (SA) model proposed by Endsley has recently drawn the attention of the research community due to its suitability for understanding the monitored environment. The SA model also facilitates an effective decision-making oriented towards the mitigation of common network problems. Hence, the present research has studied existing incident management approaches grounded in the Endsley's model, from which the three SA stages (perception, comprehension and projection) have been analyzed, with special attention on the comprehension and projection stages.

The aforementioned context has led to pose the research problem of this thesis on the adaptation of well-known feature extraction techniques, forecasting algorithms, pattern recognition methods and rule-based reasoning for designing an analysis framework targeted on granting autonomic management capabilities on self-organizing architectures. This approach is driven by use cases foreseen in emerging communication networks, which requirements must be addressed. Such use cases deal with the inference of situations that represent a likely degradation or disruption of the agreed service levels in the network.

The proper assessment of the proposed knowledge acquisition process poses an important challenge as well. This situation gains relevance due to the fact that, at the time of this writing, there are no well-known evaluation methodologies suited for 5G analytics. Furthermore, the accuracy on inferring network problems should be assessed on well-defined use cases adapted to 5G networks. For experimental purposes, special attention has been put on self-protecting capabilities, taking into account the SELFNET approach. To this end, the accuracy on detecting network incidents has been validated on two reference use cases related to Economic Denial of Sustainability (EDoS) and Distributed Denial of Service (DDoS) threats. Nevertheless, it is important to remark that the knowledge acquisition approach developed throughout this research is not limited to the self-protection domain, but to any other use case. For instance, and aside from self-protection, self-optimization, self-healing and self-configuration are the cornerstone use cases of self-organizing networks. Those could be properly embraced under the same analytical approach presented in this work, as it was showcased in the SELFNET project.

Novel network threats have also emerged through the last years, which will take another

nuance from the perspective of 5G networks. This is the case of Economic Denial of Sustainability (EDoS) attacks originated in cloud computing infrastructures, which has motivated the definition of the first use case. EDoS attacks are targeted on exploiting common vulnerabilities of the auto-scaling mechanisms with the aim to deplete the hired computational resources, thus giving rise to the economic unsustainability of the offered services. EDoS threats have drawn the attention of the research community, which led to the definition of detection and mitigation proposals. Since 5G networks are strongly dependent on virtualization platforms for provisioning services on-demand, the study of EDoS threats pose a novel research problem. Notwithstanding the importance of these attacks, the literature review disclosed two open challenges scarcely addressed. Firstly, their detection approaches are mostly grounded on network traffic metrics, hence missing the study of behavioral patterns of the auto-scaling policies in the cloud platform itself. Secondly, and to the best of the author's knowledge, EDoS detection has not been studied in the context of self-organizing networks framed in 5G architectures. Because of this, the present research explores the adaptation of novel EDoS detection methods supported by self-organizing capabilities. To this end, a formal definition of this threat and a proper validation of the use case is mandatory. These subjects have been addressed throughout this research.

Similarly, the detection and mitigation of DDoS attacks remains as an open research problem, which has lay the grounds of the second use case. DDoS has been largely studied and several detection and mitigation techniques have been proposed in the literature. However, they exhibit two aspects that should addressed for enhancing their effectiveness. On the one hand, the dynamicity of DDoS attacks in 5G networks will be characterized by the heterogeneity of the connected devices and their ability to conduct more sophisticated attacks, so raising a new concern for IT administrators. On the other hand, the possibilities to mitigate such threats are expected to be considerable higher within 5G networks, being its adaptation to non-stationary environments a distinctive trait. Aside from that, many users nowadays ask themselves if their devices could be taking part of DDoS attacks, for which detection methods based on source-side analysis of DDoS traffic plays an essential role. However, this trait has been scarcely studied by the research community and remains a research open topic. To this end, the present research explores a source-side DDoS detection approach based on knowledge acquisition by taking advantage of self-organizing capabilities. In addition, its adaptability to 5G architectures must be gounded on data heterogeneity, source traffic-flow analysis and non-stationary forecasting. Those aspects have been accounted for accomplishing the goals of this use case.

1.2 Motivation

At the time of this writing, the fifth generation of mobile networks is under development and it is expected to have its first commercial releases by the year 2020. Moreover, most of the 5G supportive technologies, such as SON, NFV and SDN, rather than reaching high maturity, are also still under development. A situation that has motivated the interest of the researchers in the last years. Such technological landscape has led to the

development of innovative and challenging projects whilst fostering high competitiveness in the academia and the industry. A fact that has been recently evidenced by the research literature.

Therefore, the research efforts committed towards the fulfillment of the first implementation of 5G have specially motivated the development of this thesis. The immersion of 5G networks is foreseen to drive outstanding changes for the society and the economy in the forthcoming years, hence relying on their capabilities to provide ubiquitous access to ultra-dense networks operating at high data rates. It is then expected by the current society to take part of such technological shift from its roots to their achievement.

From a scientific perspective, conducting research within emerging trends entails a challenging task since most of them have not reached the maturity achieved by well-known technologies. It poses a strong incentive to look for open issues where traditional formulas must be rethought or designed from scratch, which in the meantime brings higher uncertainty.

On the land of SDN and NFV, the separation of control and data planes has opened innumerable alternatives for managing complex network architectures from high-level applications. This fact has facilitated the design of novel solutions based entirely on software, disregarding low-level network aspects delegated to the data plane. In the meantime, the ability to provide self-organizing capabilities in 5G scenarios poses an additional motivation since cognitive approaches must be considerable enhanced. Those must be capable to conduct advanced analysis methods taking into account that existing computational limitations will be significantly reduced in 5G networks.

In addition, the study of well-known and novel network incidents contextualized into the 5G ecosystem embraces the definition of use cases for validation purposes, as is the case of the detection of EDoS and DDoS threats. Given the lack of existing 5G infrastructures, the development of experimental testbeds in the management side entails new challenges. Thereupon, novel 5G management frameworks, like the one provided by the SELFNET project, brought reference architectures which have inspired the proposal of self-protective capabilities for the aforementioned threats.

1.3 Objectives

The fifth generation of mobile networks has raised many challenging objectives that should be addressed in a near future, a situation that is clearly evidenced in the research landscape. This situation has led to contextualize the scope of this thesis into the area of autonomic management in 5G networks, with interest on self-organizing approaches. Therefore, this research lays out a main objective: the study of analytical methods for providing knowledge acquisition capabilities in self-organizing networks. Bearing this in mind, this work comprises three fundamental aspects. Firstly, the design of a 5G reasoning-based framework enabled for the detection of situations that are potential symptoms of degradation or disruption of the agreed service levels of the network. Secondly, the instantiation of the proposed framework into a self-organizing architecture. Thirdly, the evaluation of the detection capabilities on specific use cases framed into

emerging network environments.

For the accomplishment of the proposed objectives, the following activities shall be conducted as part of this research:

1. An in-depth study of the state of the art on 5G and supportive technologies.
2. Review of the most relevant prediction and pattern recognition methods.
3. Definition of a reasoning framework based on prediction and pattern recognition methods for acquiring knowledge in 5G networks.
4. Definition of evaluation use cases based on the analysis of well-known and novel network threats adapted to 5G self-organized scenarios.
5. Definition of high level metrics for inferring potential situations that might produce the disruption or degradation of the agreed service levels.
6. Implementation of analytic detection methods suited to self-organizing networks to address the requirements of each use case.
7. Assessment of the proposed methods when detecting network threats within the context of the reference use cases.

1.4 Contributions of this Thesis

The contributions of this thesis are illustrated in Figure 1.1. They are arranged in a matrix where the rows represent the areas of knowledge and the column identifiers stand for the contribution. The intersections denote the level of degree achieved by each contribution on the corresponding research area. Therefore, the following are identified as the key areas of knowledge: Fifth generation mobile networks (5G), Self-Organizing Networks (SON), Software Defined Networking (SDN), Cloud Computing, Prediction, Pattern Recognition and Incident Management.

To lay the background on 5G and autonomic management capabilities, contributions [SRA+16], [BWSMea17], [SMGV17] and [SMMVGV18a] have been considered. In [SRA+16], the study of the reference SELFNET architecture for providing self-management capabilities has grounded the design principles of a fully operational control loop in 5G. This approach has been considered in [BWSMea17] to rise the situational awareness adaptation for incident management in 5G networks. Bearing the autonomic control loop management in mind, a theoretical defensive approach towards crypto-ransomware mitigation has been exemplified in [SMMVGV18a]. In addition, [SMGV17] addresses the SDN principles for driving a programmable network management.

The inclusion of data analytics for achieving autonomic management in 5G has been initially studied in [SMMVGV17b] and [SMMVGV17c], leading to the integration of well-known methods in prediction, pattern recognition and self-organizing capabilities studied in [SMMVGV17d]. They have contributed for proposing a reasoning framework for 5G networks along with a detailed description of its workflows. A detailed assessment of

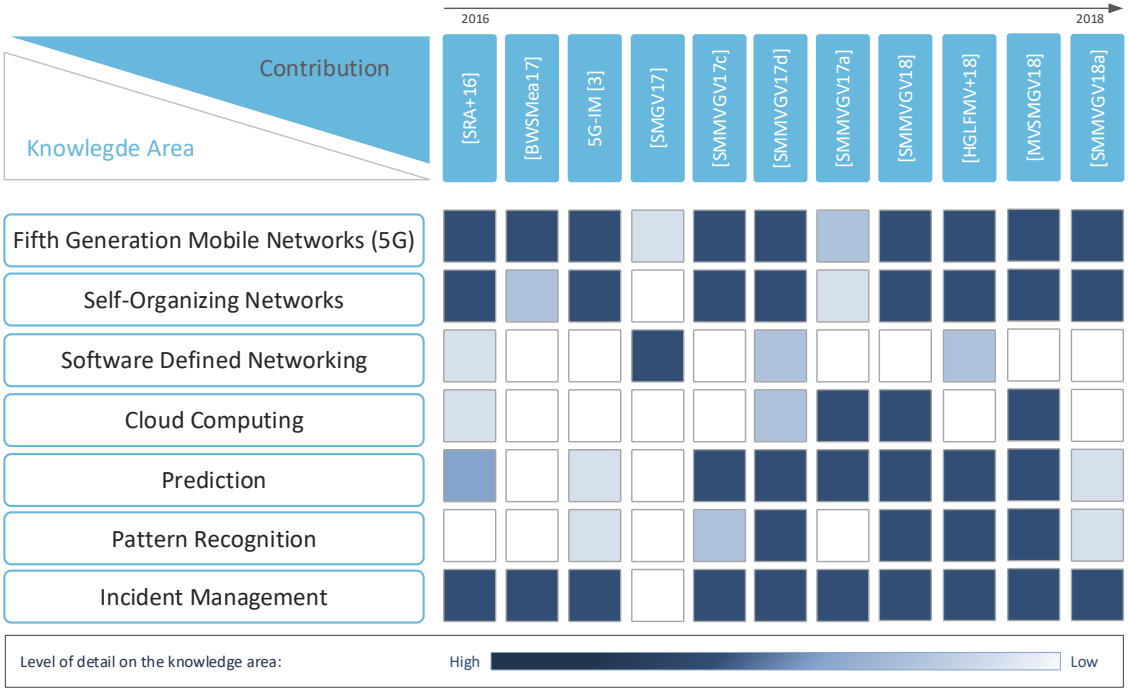


Figure 1.1: Contributions of this thesis.

its capabilities in terms of prediction and pattern recognition accuracy has been provided as well.

On the other hand, contributions in [SMMVGV17a] have led to the study of Economic Denial of Sustainability threats, which has provided an updated state of the art on EDoS attacks in cloud computing environments and the study of a novel Entropy-based Economic Denial of Sustainability detection method based on predictive capabilities. Hence, contributions in [SMMVGV18b] and [MVSMGV18] have led to the creation of a 5G security architecture with intrusion detection capabilities targeted to mitigate EDoS threats in self-organizing networks. Those contributions have grounded the proposal of a similar approach for studying the detection of DDoS attacks at source-side, which has been published in [HGLFMV+18].

1.5 Outline of the Thesis

Chapters 1 to 5 provide the theoretical background that contextualizes the research problem and related knowledge areas. They are so intended to get the reader familiar not only with the vast terminology regarding emerging networks, but also with updated research advances on 5G and their supportive technologies. On the other hand, forecasting and pattern recognition methods play a key role in the proposals introduced latter on this thesis. This fact has motivated their inclusion in the state of the art chapters.

Chapters 6 to 10 introduce research proposals aimed to accomplish the objectives stated in Chapter 1. Chapter 6 starts with the definition of a general-purpose knowledge acquisition framework contextualized in 5G networks, which constitutes the reference

architecture for addressing the use cases' objectives. They, although different in nature, share the same self-organizing approach when dealing with knowledge generation. Because of that, an in-depth study of those use cases takes place in the next three chapters. In them, the emerging network architectures and analytical methods previously introduced are referenced for sketching out the proposals and experimental validation.

In order to facilitate the reader's comprehension, this thesis has been structured as follows:

Chapter 1 introduces the research context of this thesis by examining the general landscape related with emerging network architectures, innovation fields and challenges. They allow the definition of the research problem, objectives and knowledge areas where the publications related to this thesis have contributed.

Chapter 2 reviews the state of the art of the fifth generation of mobile networks. It remarks the challenging requirements, performance indicators and generational shift fostered by their supportive technologies.

Chapter 3 introduces some research efforts towards the development of 5G networks, where the SELFNET project is described in detail. This chapter raises also important research areas that lay the grounds for the definition of novel use cases to be studied in the forthcoming chapters.

Chapter 4 reviews the main families of prediction algorithms and the adaptive thresholding methods widely referenced throughout this research. They are later studied to generate factual knowledge for prediction-based analytics.

Chapter 5 describes a set of well-known pattern recognition methods for matching, novelty detection and classification. Their ability to find discordances between samples and reference data are later applied to enhance the knowledge acquisition process from the network management perspective.

Chapter 6 focuses on the generation of knowledge about the network considering the monitored environment. To this end, a reasoning framework is proposed. It combines prediction, pattern recognition, adaptive thresholding and rule-based analysis capabilities, thus accommodating an autonomous network management approach.

Chapter 7 addresses the Economic Denial of Sustainability (EDoS) threat by delving into its definition and characteristics for proposing a detection method suited for virtualized environments.

Chapter 8 explores the implication of EDoS threats into the context of emerging self-organizing networks, for which a novel detection strategy is proposed and assessed in depth.

Chapter 9 delves into the source-side detection of Distributed Denial of Service (DDoS) threats in self-organizing networks. To this end, a novel detection method to infer the participation of a device in a DDoS attack is studied.

Finally, Chapter 10 summarizes the contributions derived from this research and raises some interesting lines of future work.

Chapter 2

5G: Fifth Generation Mobile Network

This chapter presents a general overview of the fifth generation of mobile networks, their challenges, requirements, Key Performance Indicators (KPI), and future trends. For a better understanding of the reader, the chapter is organized in five sections. In Section 2.1, a broad picture of 5G network is presented. Section 2.2 describes requirements pushed towards the development of 5G networks. Section 2.3 presents the main Key Performance Indicators related with the accomplishment of 5G requirements. In Section 2.4, the main 5G supportive technologies are introduced. Finally, Section 2.5 provides the final remarks of this chapter.

2.1 Overview to 5G

The growth of mobile devices connected to Internet has increased the number of user necessities and, in consequence, the network requirements. Real challenges have raised in terms of network performance indicators such as faster recovery times, lower latency, higher transfer rates, less energy consumption, enhanced Quality of Service (QoS)an Quality of Experience (QoE), among others.

It is expected that in the year 2020 the number of connected devices will exceed 20 billion [Gar17]. Hence, the main telecommunications operators are pushing novel technologies in order to enhance the current network characteristics torward the fifth mobile generation. 5G has thus emerged as the promising technology to deal with the challenging requirements of future generation networks [NGM15]. It empowers a smart integration of the most innovative advances on Network Function Virtualization (NFV), cloud computing, Software-Defined Networking (SDN), Artificial Intelligence (AI) and Self-Organizing Networks (SON). In particular, the synergies between SDN and SON are addressed as a key research topic [Eur14] to enable the fulfillment of the disruptive key performance indicators (KPIs) promoted by 5G [GRA16]. On this line, several research initiatives in 5G have integrated cloud computing, SON and cognitive networks to design modern and automated management architectures which are able to monitor heterogeneous network infrastructures. That is achieved by the incorporation of analysis,

decision-making and learning capabilities when inferring the status of complex network scenarios, so the automatic enforcement of actions to mitigate risks or optimize operations is possible [ABC+14].

5G will take advantage from the synergy of such technologies in order to provide autonomic self-management. For this purpose, 5G should grant novel capabilities on context modeling and network programmability to accomplish higher QoS/QoE levels, more efficient operational (OPEX) and capital (CAPEX) costs, enhanced network trust and privacy, and many other distinctive features [SYY+13][PZCG14]. In the mantime, this mobile generation will be driven by economic, societal and operational trends on which ensuring the network Service Level Agreements (SLAs) will be even more challenging.

Furthermore, 5G environments will have to face crucial issues related to scalability [DMR16]. In this context, cloud computing can deal with these concerns by provisioning unlimited resources (computing, storage and networking) when the service is needed. On the other hand, Mobile Edge Computing (MEC) allows the deployment of Virtual Application Functions (VAFs) such as critical applications, location services, data caching, among others, closer to the edge location for granting the user higher bandwidth and lower latency when the service is delivered. One of the main advantages of both cloud computing and MEC is that virtual resources are deployed timely and with minimal effort. In this way, the telecommunication providers can turn into a more profitable and efficient business model benefited from a significative reduction on the operational costs.

A key aspect of 5G networks is the introduction of advanced data processing techniques. In this regard, specialized analysis tasks and artificial intelligence methods should be properly applied to generate knowledge based on the information gathered from the network. Such advanced analysis relies on aggregation and correlation techniques applied on low-level metrics acquired from both traditional monitoring tools and advanced network sensors.

2.2 5G Requirements

In order to cover the main necessities of modern mobile networks, 5G requirements has been divided in some application domains [NGM15][5G-16a], as is detailed in Table 2.1.

One of the main concerns is related to the number of devices/sensors connected to 5G networks, a situation that becomes more complex considering the fact that the Internet of Things (IoT) poses extra requirements that 5G must be able to cover. Such complexity is raised by the user expectations on perceived quality, and even more by the business model focused on agile service delivery. On this line, 5G demands additional efforts to be undertaken by the research and industrial communities.

In general terms, 5G architectures must provide enhanced functions in order to cover context life cycle [PZCG14], which can include the following phases:

- *Context acquisition.* This phase gathers information from heterogeneous sources, including physical and virtual elements. It considers not only the context data source but also the frequency, interfaces, data processing, formats, and the acquisition method.

Table 2.1: Summary of 5G Requirements

Requirement	Description
User experience	This requirement is related with the enhancement of the quality of service (QoS) and Quality of Experience (QoE) levels compared with current networks [TPL+16]. On one hand, this requirement can be achieved by the combination of MEC and SDN technologies. On the other hand, the deployment of virtual functions over a virtualized cloud environment closer to the service location will let a better user experience.
Device	This requirement improves the intrinsic device indicators such as higher data rates, signal and energy efficiency, among others. An increase of 1000x times of mobile traffic is expected by 2020 and higher QoS levels must be reached on 5G infrastructures. Besides that, it is important to promote advances in device to device communication (D2D); monitoring, aggregation and processing capabilities; extend battery lifetime and programmability at both software and hardware elements [BTAS14].
System Performance	This requirement aids to cover advanced capabilities to support higher data transfer speed, more simultaneous connections and "zero perceived" latency. An estimated data rate is 10Gbps.
Enhanced Service	The main idea behind this requirement is to enhance the user experience by granting higher availability, reliability and accuracy of the device location. Likewise, transparent connectivity and enhanced security services will offer seamless connectivity and enhanced privacy. It includes self-protection and self-healing capabilities in order to improve network resilience.
Business Model	This requirement is intended to introduce new business models in order to customize the network behavior while reducing operational and capital cost. 5G will be able to introduce new services (time to market) in a faster and dynamic way, taking advantage of the separation between of data and control plane, the deployment of virtual functions and the possibility of sharing network resources.
Management & Operation	Current networks require a high degree of manual management and configuration of network devices, so that increasing the operational costs. In this regard, the automatic management requirement will allow the reduction of these expenditures by deploying new services anytime and anywhere when demanded by the users. A key characteristic of future 5G networks is the ability to perform automatic system recovery and self-management in critical situations, hence minimizing also the Total Cost of Ownership (TCO).

- *Context modeling.* The collected network data is represented in a specific model which must have a meaning taking into account the context of the situation. That collected data needs to be characterized in terms of its attributes or characteristics, using different approaches such as ontology methods, markup schemes, graphical representations, object or logic oriented modelling.

- *Context reasoning.* This phase obtains high level metrics from the collected data, thus helping in the creation of new knowledge. For this purpose, the modelled

data will be analyzed by reasoning methods and event management techniques. The knowledge inference includes preprocessing, aggregation and analysis of the acquired data.

- *Context Distribution.* Finally, all information (raw data, aggregated and high-level metrics) are delivered to stakeholders with the objective to enhance the provisioning of their services.

2.3 Key Performance Indicators (KPI)

5G networks pose diverse Key Performance Indicators (KPIs) in order to measure the expected services in comparison with the current mobile technology. Those indicators can thus be defined in terms of the following metrics [Eur14] [NGM15]:

- *Latency.* 5G expects to provide lower latency compared to current networks. It takes into account the latency between the source and destination, that is knowing as end-to-end latency. This KPI expects to enhance the quality of experience of the user. It is foreseen to reach 5ms.
- *Capacity.* 5G will be able to support higher capacities (volume of information) not only with normal traffic but also when many users are connected in the same geographical area. The referential value is expected to reach 10 Tbps/Km2.
- *Service creation time.* One of the main concerns in current networks is the operational costs when a new service is required. In this regard, 5G expects to reduce the service creation time from 90 days to 90 minutes.
- *Number of connected devices.* In order to achieve a future network landscape with any connected devices, this KPI aims on increasing the number of devices in 1000x. An estimated value is about one million per square kilometer (1M/Km2).
- *Energy efficiency.* 5G will decrease the energy consumed by network infrastructures in 90% compared with current technologies.
- *Location accuracy.* 5G expects to provide a location accuracy of about one meter (1m). This value takes importance for industrial infrastructures mostly related with transportation and positioning systems.
- *Mobility.* This KPI will allow to enhance a continuous access to the services even in high mobility conditions. To this end, 5G expects to provide mobility speeds of about 300 to 500Km/h.
- *Peak Data Rate.* This KPI is related to data transfer rate reached by network devices, with an expected value of 10Gbps.

In Figure 2.1 the expected value or reduction of each KPI is presented. It is important to note that some of these values are represented taking into account the capacities of current mobile networks.

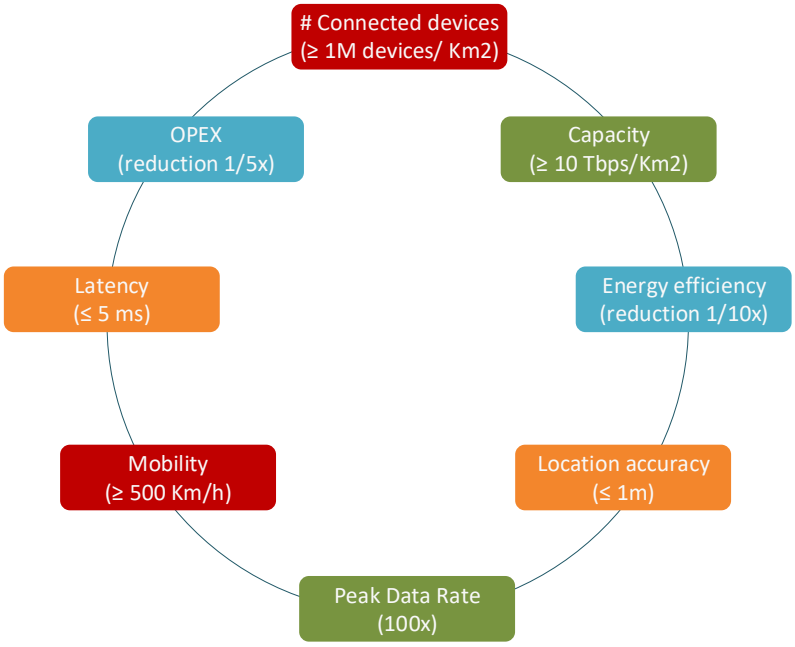


Figure 2.1: 5G Key Performance Indicators.

2.4 5G Supportive Technologies

In this section, the most outstanding 5G supportive technologies are revised. SDN, Self-Organizing networks, Network Function Virtualization and Cloud Computing are introduced since the thesis contributions are mainly grounded on them.

2.4.1 Software Defined Networking (SDN)

In 2011, a group of network operators, service providers and industry representatives created the Open Networking Foundation (ONF) [ONF], an organization that promotes the adoption of SDN in the industry. ONF defines Software Defined Networking (SDN) as a network architecture where the control is programmable and is separated from the packet forwarding functions, inherent to lower levels of the OSI model. In a more extended context, SDN is an architecture that optimizes and simplifies network operations, linking the applications with network services and devices. For this purpose, a centralized logical control entity, called the SDN controller, is used to coordinate the communication between the applications and network elements. The controller exposes network functions and operations through Application Programming Interfaces (APIs), leading to a suitable interaction with the SDN applications [NG13].

OpenFlow [MAB+08] is the main protocol that has facilitated the development of SDN, and has been adopted for the industry and research community. OpenFlow is the first standard communication interface defined to communicate the control and infrastructure layers of the SDN architecture. The most widely adopted version is 1.0.0.

2.4.1.1 SDN base architecture

Figure 2.2 depicts a logical view of the SDN architecture [ONS12], composed by three layers: infrastructure, control, and application. In the infrastructure layer, network devices (switches and routers) forward packets according their configured flow tables. The control plane contains the SDN controllers responsible to configure forwarding rules in the flow tables of each network device existing in the infrastructure layer. In the upper layer the different SDN applications are located. The architecture layers are communicated trough programming interfaces that allow their interaction and information exchange. Network intelligence is logically centralized in the SDN controller, which is able to maintain a global view of the network. The separation of network functions, in control and data planes, is a distinctive characteristic of SDN. The role of each plane is described in the following two sections.

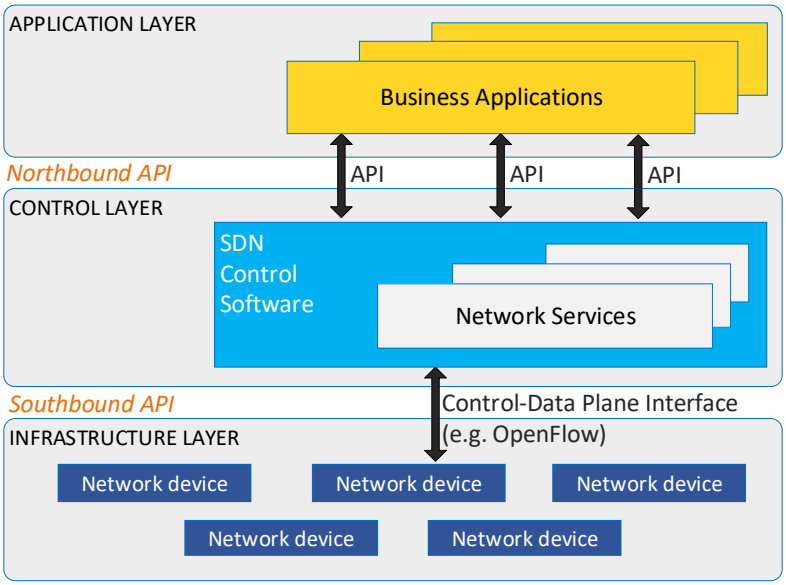


Figure 2.2: SDN Architecture.

2.4.1.2 Control Plane

The control plane is responsible to set up the logic and corresponding data sets necessary to control the SDN network behavior. Network protocols routing rules (as the ones of OSPF) or filtering policies in a firewall are some examples of control plane logic in IP networks [NG13]. The main objective of the control plane in SDN is the management of flow tables to define forwarding rules. To accomplish this, a global abstraction of the network is mandatory. This logical abstraction is programmable and is performed through a Network Operating System (NOS), by the use of any protocol enabled to get information from the data plane through a software interface [FRZ14] (also called Southbound API), such as OpenFlow.

2.4.1.3 Data Plane

The data plane is responsible to process incoming network packets in the network devices. The packets get into the devices through physical media (cable, optical fiber, and so on). After being assembled and error-checked, well-formed datagrams are processed in the data plane through flow table lookups, previously configured by the control plane [NG13]. The only exception to this procedure arises when a datagram does not match with any flow table entry. In this case, the packet is sent to the controller, which processes it according to the SDN applications logic. IP packet routing and layer two switching functions are some examples of data plane functions in IP networks [FRZ14]. Infrastructure elements include not only network switches, but also terminal devices.

2.4.1.4 OpenFlow Protocol

This interface allows the SDN controller to directly configure the forwarding flow tables in the network devices, such as routers or switches that support this protocol. ONF is responsible to standardize this protocol. In OpenFlow, it is mandatory to set matching rules in the flow table entries that will be examined for every incoming packet in an OpenFlow switch, in order to decide the corresponding action to take over the packet (i.e. forward, drop, etc.). Figure 2.3 shows the packet header fields used to define matching rules, according to OpenFlow version 1.0.0 [OFv].

Ingress Port	Ethernet Source	Ethernet Destination	Ethernet Type	VLAN ID	VLAN priority	IP Source	IP Destination	IP Protocol	IP ToS bits	TCP/ UDP src. port	TCP/ UDP dst. port

Figure 2.3: Packet fields used to match against flow table entries.

2.4.1.5 Centralized vs Distributed approach

IP networks have been built under a distributed scheme in which every network node has its own instances of the control and data planes. By contrast, SDN consolidates the whole network control plane and provides a centralized scheme that determines its behavior. Figure 2.4 depicts a comparisson between the two aforementioned schemes.

The separation of planes in SDN allows a continual evolution of each independently. On the one hand, the hardware over which SDN is implemented, the embedded data plane software and interfaces, and, on the other hand, the development of network operating systems and high-level applications that implement the best software engineering practices. These advantages give SDN the flexibility that traditional network architectures cannot offer.

2.4.2 Self-Organizing Networks (SON)

From a functional perspective, a SON network includes three outstanding capabilities: self-healing, self-optimization and self-configuration [Nom08]. Self-configuration involves

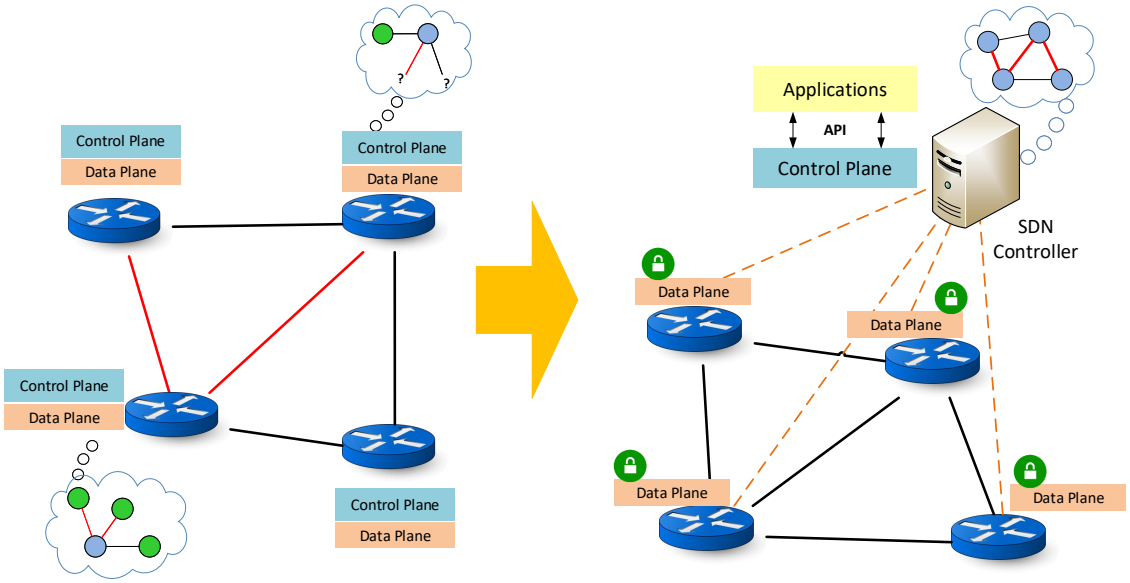


Figure 2.4: From the distributed scheme in IP to the SDN centralized scheme.

the processes to automatically configure new network nodes deployed in the network by downloading and parameterizing the required software [PLW+13]. Self-optimization addresses optimal performance conditions by comparing the current network status with the target parametrization, which might trigger corrective actions when required [AAA14]. In addition, self-healing entails the detection of network failures, the diagnosis of the situation, and the enforcement of recovery actions to restore the affected network function [AIIE13]. The original SON architecture [Nom08] comprises an Operation, Administration and Maintenance (OAM) subsystem deployed either as a centralized, distributed or hybrid architectural approaches. The simplest closed-loop network OAM architecture [HSS12a] entails the presence of sensors deployed along the network to monitor crucial network elements for Performance Management (PM). The monitored information is analyzed according with the network policies defined to automate the Configuration Management (CM) processes. Such policies are planned and enforced through different network actuators adapted to specific domain managements [SK09]. Figure 2.5 illustrates a simplified view of a closed-loop architecture.

Furthermore, Glenn et. al. [AIIE13] outline the differences between adaptive, autonomous and cognitive networks. Adaptive networks change their configuration in response to environmental changes, whereas autonomous networks are also adaptive, but with no human intervention. Thus, SON networks are adaptive and autonomous. On the other hand, cognitive networks are autonomous networks that include learning capabilities in the management lifecycle; ranging from monitoring, decision-making and action enforcement processes in order to acquire knowledge about the context [JVDMB+07]. Thereby, SON-cognitive networks have represented an ongoing trend for the management of modern networks [FM09], in this way adapting the processes related with sensing the network, planning, decision-making and actuation according to the operational context.

Even though the original SON definition remains suitable, it has adopted new

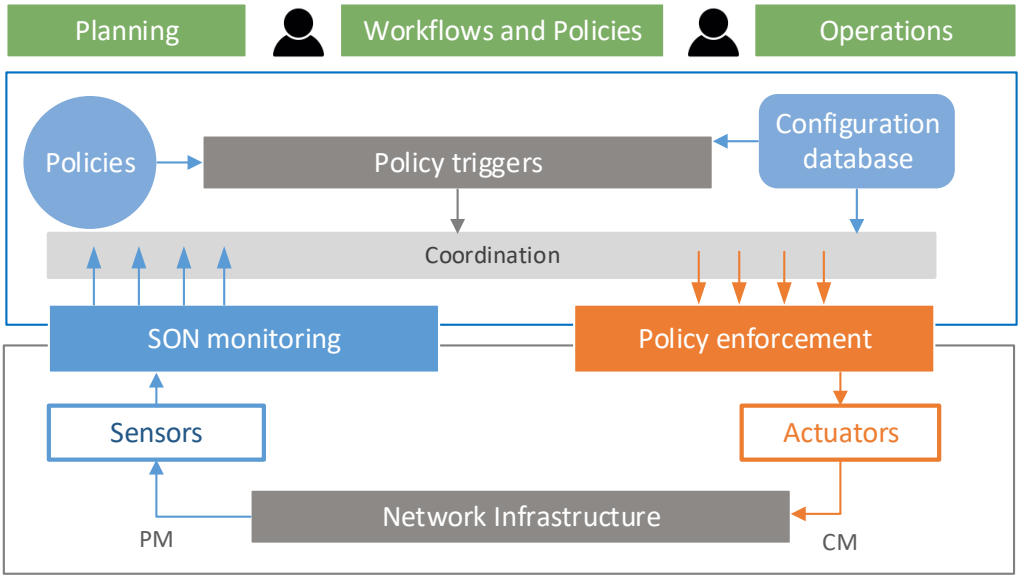


Figure 2.5: Closed-loop Self-Organizing Network.

characteristics adapted to the evolving mobile network technologies such as LTE, 4G and 5G. In this context, emerging SON networks that include cognitive capabilities cope more efficiently with the management challenges pushed by the operators. Modern sel-organized scenarios grant the ability to perform complex network analysis targeted on deploying smart proactive and reactive actions when mitigating, correcting or optimizing network services. SON networks are key enablers of 5G, taking advantage of the application of advanced machine learning techniques to adapt the network for complex self-healing, self-optimization, self-configuration, self-protection and other use case scenarios. 5G provides suitable capabilities to evolve more powerful SON architectures relying on the decoupling of network and data planes promoted by SDN, the advanced on-demand provisioning fostered by cloud platforms, the immersion of NFV management and orchestration architectures, and the evolution of self-managed approaches towards the next generation networks.

2.4.3 Network Function Virtualization (NFV)

The provisioning of efficient network services with higher quality requirements and lower operational costs has been traditionally a major concern for service providers in the telecommunications landscape. The bundling of network functions within closed hardware platforms have limited the ability to extend the capacity of the offered network services not only by budget constraints, but also for the lack of agility in the deployment process [NG13]. Moreover, when considering the factors involved in the provisioning of network services towards an efficient relationship between revenue and CAPEX/OPEX (i.e skilled network operators, vendors, time-to-market, users demands or hardware/software updates), higher infrastructure investments are mandatory. However, some studies have shown that their return rate of such investments is minimal [HSMA14], hence demanding more innovative business models.

On the other hand, the concept of virtualization has emerged to allow hardware resource sharing between software entities that operate independently, as if they were completely isolated from each other, thus managing the workload more efficiently [MAJ14]. This is why the advantages of virtualization have been adopted on the networking area for allowing the coexistence of multiple logically separated networks running on the same physical elements [CB10]. Thus, enabling the abstraction of the hardware infrastructure by standard software interfaces for a proper network management.

Bearing in mind the restrictiveness of hardware-based solutions, Network Function Virtualization (NFV) can be defined as the bundling of specialized network applications such as firewalls, load balancers, DPI servers, among others, into software entities rather than in hardware devices [HSMA14]. Provided by the flexibility in the management of software-based solutions, a service provider can deploy customized services adapted to the monitored network conditions by placing network function instances (VNFs) of the desired services in different locations of the network. However, to accomplish an on-demand service provisioning, advanced orchestration processes are mandatory for ensuring the lifecycle of VNFs across the managed premises.

In 2012, a group of telecommunication operators of the European Telecommunications Standards Institute (ETSI) proposed the NFV framework [ETS13]. It lays the architectural vision and design considerations for virtualizing network functions on the supporting infrastructure, and defines also the communication interfaces for the different components, as illustrated in Figure 2.6. The framework relies on standard Commercial off-the-shelf (COTS) hardware to set the architectural baseline for orchestrating the allocation of network functions throughout a virtualized infrastructure.

Basically, the reference architecture defines three main working domains: The Network Function Virtualization Infrastructure (NFVI), Virtualized Network Functions (VNFs), and NFV Management and Orchestration. The framework might also integrate operational/business support systems (OSS/BSS) used by network operators for management purposes [ETS13].

NFVI comprises hardware and software elements that lay the virtual platform on which VNFs are created. NFVI is composed of compute, storage and networking components abstracted from their counterpart hardware [YZV16]. This level of abstraction is achieved by the Virtualization Layer, since it can expose the virtual platform as a single entity with disregard of the physical placement or topology of hardware elements in the managed domain. Thereby, VNFs can be deployed in different locations of the network, provided by the proper allocation of compute, network and storage resources [CB10]. On top of the VNFs, the Element Systems (EMs) are intended to manage the functionality of one or many controlled VNFs.

In addition, the NFV Management and Orchestration implements the coordination between the architectural components of the framework in such way that the VNFs provisioning lifecycle is guaranteed when delivering a network service. It poses a strong dependence on Virtual Infrastructure Manager as the controller of the abstracted compute, storage and network resources to meet the deployment requirements of each VNF in the provisioning process. Even though the NFV framework allows the dynamic allocation

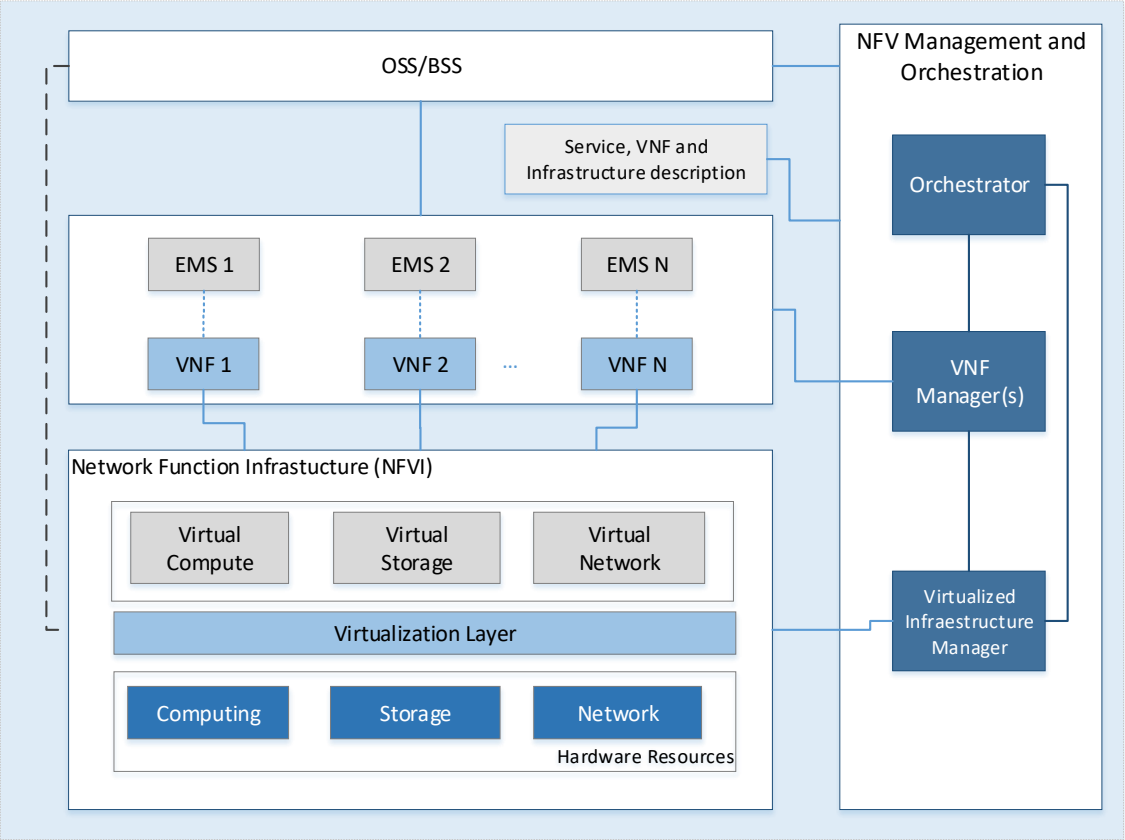


Figure 2.6: ETSI-NFV Architecture.

of network functions, the placement for VNFs chaining in datacenters is a challenging topic from the network operator point of view, which requires optimization strategies for dealing with resource consumption, cost evaluation, quality of service levels, and so on, when managing large infrastructures [XSZ+15].

It is also worth mentioning the tight integration achieved by NFV and SDN in emerging network scenarios. Even when both technologies can coexist independently, their synergies have opened a wide range of opportunities for network operators. SDN/NFV have introduced programmable management approaches in the deployment of network services suited for challenging use-cases including, for instance, video services or the "cloudification" of telco infrastructures, which have been largely addressed by the research community [CRSG+15] [BKH+14].

2.4.4 Cloud Computing

Among the different definitions of cloud computing, the one stated by the National Institute of Standards and Technology (NIST) has been widely referenced in the literature [ZCB10]. It defines cloud computing as a model for enabling ubiquitous, convenient and on-demand access to a shared pool of computing resources that can be timely provisioned with minimal management effort or service provider intervention [NIS11].

With the rapid evolution of information technology architectures and enterprise

business models, traditional provisioning of resources (computing, storage and network) has demanded significative advances to deal with the limitations of hardware computing elements. In this respect, virtualization has played a major role as it allows sharing infrastructures transparently from the customer point of view [MAJ14]. In cloud computing, a particular resource can thus be leased to different users while ensuring a complete isolation from each managed domain, a concept known as multi-tenancy. This innovative service-oriented model provides an elastic resource provisioning model, while decreasing the costs of hardware acquisition. Cloud computing offers in this way different service models depending on the available resources to be leased [DLNW13]. Zhang et. al [ZCB10] outlines also some related technologies that led to the emergence of cloud computing.

Previously, grid computing was aimed to merge computational capacity for achieving a specific goal. That model was further exploited by the cloud computing paradigm which provides a general-purpose virtualized infrastructure for resource sharing. Such enhancement is accomplished by the virtualization engine (hypervisor), capable of abstracting the underlying hardware elements. On the other hand, concepts like utility and autonomic computing have grounded the cloud approach. Utility computing aims on billing customers according to their resource usage; whereas autonomic computing has the objective to grant self-management capabilities to a system, being this a feature from which cloud computing and modern technologies, such as 5G, have benefited from.

The NIST has also identified the following five essential characteristics in cloud computing deployments [NIS11]:

- *On-demand self-service.* The automatic provision of resources as needed by a particular customer.
- *Broad network access.* Cloud capabilities available over the network for being accessed by standard mechanism.
- *Resource pooling.* The ability to pool resources for serving multiple tenants for which the sense of location is abstracted for simplicity (i.e. at country, state or datacenter level).
- *Rapid elasticity.* It is understood as the elastic provisioning capability achieved by means of scaling mechanisms. From the customer perspective, resource scaling is unlimited.
- *Measured service.* The inclusion of a resource metering per service type that let a cloud provider the abstraction of usage profiles (i.e. for billing purposes).

Aside from their definition and characteristics, the service model is driven by the cloud multi-layered architecture [ZCB10][DLNW13] illustrated in Figure 2.7. Their architectural components are explained as follows:

Infrastructure as a Service (IaaS) provisions infrastructure resources on-demand, which are typically categorized as compute, storage and networking. Datacenter physical

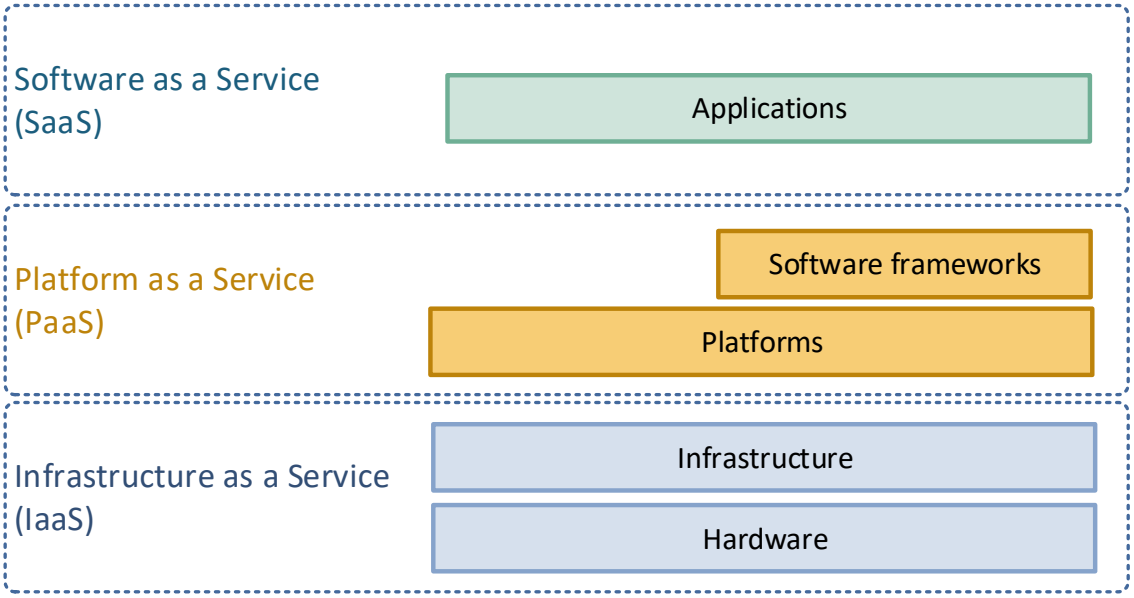


Figure 2.7: Cloud computing architecture.

entities such as servers, routers, power systems, etc. are pooled and abstracted by the virtualization layer to grant the provider the ability of offering on-demand infrastructure resources. Commercial IaaS platforms are Microsoft Azure and Amazon EC2.

Platform as a Service (PaaS) is the intermediate level of the architecture. It offers an integrated environment built on operating systems and application frameworks for creating, developing and deploying custom applications. PaaS platforms examples are Google App Engine or Amazon S3.

In the top level, *Software as a Service (SaaS)* consists of software applications delivered directly to the end users on the Internet. It allows them a "pay-as-you-go" model derived from the use of the hired applications. Google Docs is an example of this.

Since most of the cloud providers are private, the research community, industry and enterprises have also dedicated efforts towards the creation of open source projects aimed to deploy customized cloud environments with no dependence neither on external parties nor proprietary hardware/software elements. In particular, Openstack [Ope] has emerged as a leading open source platform for managing cloud deployments. Openstack orchestrates a set of services (Nova, Neutron, Keystone, Telemetry, etc.) which could be also integrated with SDN and NFV deployments.

Hence, in the context of emerging network platforms, cloud computing facilitates the on-demand network resource provisioning by means of the elastic capacities inherent to their supportive platforms.

2.5 Final Remarks

In this chapter the principles of 5G networks have been explored. It has started from a description of their challenging indicators to a later reviewing of their supportive technologies. One of the most important concepts to remark is the idea behind

programmable networks, on which the separation of the data and control planes has shifted the network management paradigm towards a software-driven model with self-organizing capabilities. Likewise, the scalability and flexibility achieved by the virtualization of network functions has led to evolve the network infrastructures towards a software-based service model. As it has been seen, the characteristics of all reviewd technology enablers will be seamlessly integrated in the 5G ecosystem for fulfilling the emerging connectivity demands. Hereinafter, and bearing in mind that 5G is currently under development, the role of those technology enablers has been considered throughout the performed research.

Chapter 3

Research in 5G and Related Open Challenges

This chapter describes the current state of the research landscape related to 5G technologies. Challenging directions are also presented by reviewing current research projects carried out to address a variety of use cases. This chapter focuses also on the study of network incident management and its adaptation to emerging 5G scenarios. For this purpose, two types of attacks are reviewed as potential use case scenarios to be addressed by 5G networks, being them furtherly studied later in this document. The contents of this chapter are structured in four parts. Section 3.1 presents some initiatives conducted by the research community to promote the advances on 5G technologies. Section 3.2 introduces the SELFNET project in more detail due to its close relationship with the research conducted on this thesis. Section 3.3 describes the traditional network incident management concepts and its relationship with the 5G scenarios. Finally, Section 3.4 provides the concluding remarks of this chapter.

3.1 5G research projects

Research initiatives in 5G have been conducted to cope with the challenges of new generation mobile networks. These initiatives are worldwide fostered and funded by different organizations, such as the European Horizon 2020 programme [ECH] (preceded by FP7), the IMT-2020 group in China [RPI], 5G Americas [R5A], among others.

In the European context, some FP7 projects such as METIS [OBB+14], T-NOVA [PTn], UNIFY [RPU] and COWD [CrF] have contributed to lay important research baselines for other initiatives. For instance, the Mobile and wireless communications Enablers for the Twenty-twenty Information Society (METIS) generated and agreed an European platform for the development and standardization of mobile and wireless communication systems. Likewise, T-NOVA takes the advantages of SDN and NFV thechnologies, being them focused on the automated deployment of Network Functions as a Service (NFaaS) over virtualized infrastructures. To this end, the projects aims on the design and implementation of a platform enabled for the provisioning, configuration, monitorization and optimization of virtual functions.

Complementary, a dynamic platform for automated end-to-end service delivery is explored by the UNIFY [RPU] project, which targets the dynamic and orchestrated provision of services in cloud environments with optimal placement of service components across a novel infrastructure. Unlike T-NOVA and UNIFY, which rely on the potential of SDN, NFV and cloud technologies, the CROWD project is intended to significantly increase wireless mobile network density; ensuring user quality of experience, resource optimization and a reduction on energy consumption.

Promoted under the 5G-PPP Phase 1 projects umbrella, some outstanding H2020 initiativesare METIS II, COGNET, CHARISMA, 5G-ENSURE and SELFNET. They encompass several research fields and, at the same time, promote collaboration partnerships to enhance current and future outcomes. The METIS II [RPM] project aims to develop a seamless integration of 5G radio technologies by the insertion of a protocol stack architecture addressing regulatory and standardization challenges. It provides a collaboration framework for 5G as well. A smart management of the Cloud Radio Access Network (C-RAN) is addressed by CHARISMA [PCM] project, leading to the efficient deployment of network services through the intelligent management of C-RAN deployments and the Radio Remote Head (RRH) platforms. CHARISMA targests on achieving low latency, higher density, increased data rates and spectral efficient and enhanced energy management. On the other hand, incident management challenges in 5G are targeted by the 5G-ENSURE [R5G] project, which covers a wide range of security and resilience concerns like those dealing with standardization, privacy and architectural aspects. 5G-ENSURE has the final goal to provide reliable security services with "zero perceived" downtime.

Network Functions Virtualization and service chaining are a disruptive capability in 5G. In this regard, the SONATA [RS2] project addresses the challenges related to their development and deployment. To this end, it enables a Software Development Kit (SDK) and an orchestration framework to offer a platform for the rapid provisioning of services and applications. A machine learning approach to allow autonomic network management is proposed by the COGNET [P5C] project, which attains self-organizing capabilities based on the monitored information. This approach is conducted by machine learning algorithms applied to identify network errors, fault conditions, security or other related issues that led both inferring the network context and predict the user demands. As a result, the provisioning of services is dynamically adapted to the inferred network context. A similar approach is conducted by the project SELFNET [P5S], which evolves the concept of self-organizing networks in 5G by developing a framework for autonomic network management. SELFNET is further explained in the next section.

Several other research efforts are described with more detail in [BLVCSMGV16] and [Pir14]. All of them share the common vision to fulfill the proposed 5G KPIs, hence enabling a feasible adoption of the fifth generation of mobile networks.

3.2 The SELFNET project

SELFNET (Self-Organized Network Management in Virtualized and Software Defined Networks) was proposed with the main goal to develop a smart and autonomic network management framework to provide self-organization capabilities in mobile 5G networks. This project has been funded by the Horizon 2020 programme.

SELFNET achieves an autonomic management paradigm by the integration of enhanced monitoring features, prediction algorithms, pattern recognition strategies, machine learning capabilities, orchestration of virtual functions and service function chaining to conduct a self-organized approach. Such paradigm is focused on the identification of the current network behavior, which leds to decide the best mitigation responses against the inferred network problems. In consequence, the deployment of proper actuators in the infrastructure takes place.

3.2.1 SELFNET Architecture

SELFNET reference architecture [NCC+16] relies on the principles of SDN and NFV to allow an intelligent management of different network functions intended to detect and automatically mitigate a range of common network problems such as network congestion, transmission delays, link failures, among others. SELFNET defines three major use-cases: self-protection [NCC+16], self-optimization [NWAC+16] and self-healing [SRA+16]. Each of them distinguishes several scenarios in which network analysis, decision making and action enforcement are required.

The architecture (Figure 3.1) is composed by functional layers, each of them described as follows:

3.2.1.1 Infrastructure Layer

In the bottom part, the Infrastructure Layer holds the physical resources required to deploy and instantiate virtualized functions. To this end, it is further divided into sublayers for provisioning physical computing, networking and storage support over bare metal, and a virtualization sublayer capable to instantiate the required virtual infrastructures for supporting VNF deployments. Cloud computing platforms play an important role towards the consecution of advanced network abstraction and elastic resource provisioning.

3.2.1.2 Virtualized Network Layer

The Virtualized Network Layer holds the instantiated network infrastructure that support the execution of virtualized network functions (VNF) both individually and chained. In the latter case, being composed as Network Services (NS) deployed along the virtual topology.

3.2.1.3 SON Control Layer

In the next level, the SON Control Layer contains the different SON sensors for data collection to start the intelligence loop, and the SON actuators for the enforcement of actions previously decided for closing the control loop in the managed network domain.

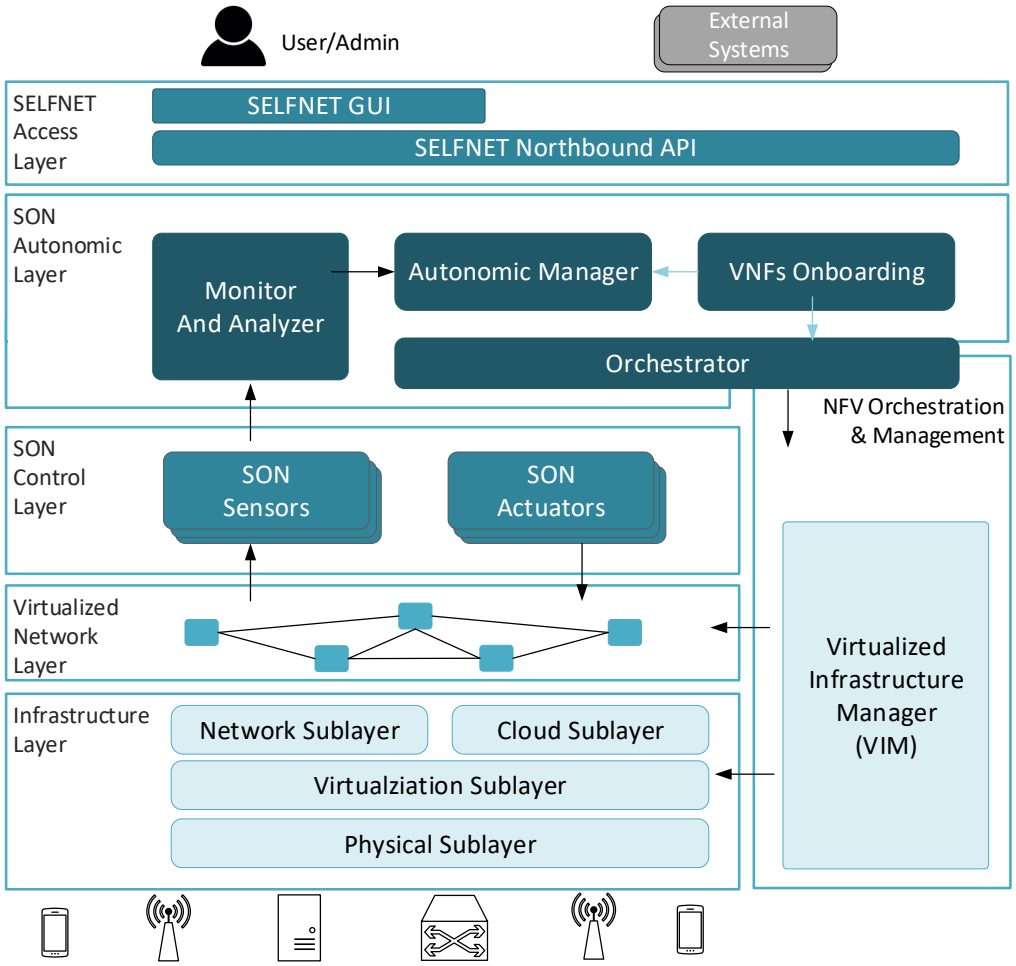


Figure 3.1: The SELFNET Project Architecture.

3.2.1.4 SON Autonomic Layer

The upper level is for the SON Autonomic Layer, which is the core of SELFNET intelligence and it is committed to monitor relevant data for acquiring knowledge about the monitored network. The obtained knowledge leads to both diagnose the cause of potential network failures and take decisions on the best actions to be enforced. Hence, accomplishing the system goals and the compliance of the agreed service levels.

3.2.1.5 NFV Orchestration and Management Layer

A supportive architectural component is the NFVO Orchestration and Management Layer, which allows automated and efficient orchestration mechanisms for deploying network functions in the network infrastructure. The main goal of this supportive module is the management of the VNFs lifecycle, accommodated to the service provisioning needs.

3.2.1.6 Access Layer

In the highest level, the Access Layer allows the interaction of SELFNET with external users, network administrators or external systems through suitable Application

Programming Interfaces (APIs) for an efficient management of the system.

3.2.2 Situational Awareness in SELFNET

To facilitate the operational context comprehension based on the Endsley situational awareness model [End88], a three-step schema of Monitoring, Aggregation and Correlation, and Analysis is proposed. This, in fact, maps the Monitor and Analysis component of the SON Autonomic Layer:

- *Monitoring* has the main objective of collecting a wide range of low level metrics and events from the physical and virtual network infrastructure, and from the deployed SELFNET sensors.
- *Aggregation and Correlation* methods reduce the amount of the monitored information, by obtaining aggregated metrics about a specific network domain. Meanwhile, events are correlated and filtered to avoid redundant or non-sensitive information.
- *Analysis* is aimed to acquire knowledge about the operational network context inferred from the analysis of aggregated monitored metrics. This process is carried on by pattern recognition capabilities, prediction methods, and knowledge inference procedures to deduce conclusions regarding potential network failure or degradation scenarios projected from the observations.

3.3 Network Incident Management in 5G

In the last decades a great variety of contributions related with the management of incidents have been published. Some of them are classified and reviewed in depth in [SRA+16] [SSABC16], where a marked trend toward the adoption of classical information security risk management schemes is emphasized. They pose different lines of research, ranging from the mere definition of the risk management terminology [HA14], to the proposal of practical guidelines for their mitigation [Dot15].

The first of these incident management groups of publications lies in the foundation of conceptual security models, as is the case of the well-known CIA-Triad [Smi12], the Parkerian Hexad [PP98] or the Cube of McCumber [McC91]. On the other hand, several authors focused on the study of how mitigate potential threats, hence establishing the basis for standards [Int13] [NSP], guidelines [ReC] or platforms [MMA]. As indicated by Ben-Asher et al. [BAG15] a greater specialization in this area of research and its applications representatively improves the effectiveness of defensive deployments, which is a very important step towards bringing self-organizing capabilities on 5G environments. But it is important to highlight that in the networking context, an incident does not only report a risk. In fact, they occur in connection with something else, which may be the result of a security threat, but also the outcome of the deployment of countermeasures, or even the variation of certain network management policies, such as enabling additional bandwidth, the discovery of new devices or the optimization of some resource usage, in

this way also bringing feedback about the effectiveness of the self-management actions. Therefore, understanding the nature of an incident and its impact usually requires a comprehensive overview of the state of the network and the different cause-effects monitored in them. Because of this, most of the recent proposals for network incident management combine the conventional risk management schemes with the situational awareness model proposed by Endsley [End88].

According with Endsley, situational awareness means "to have knowledge of the current state of a system, understand its dynamics, and be able to predict changes in its development". Because of this, the model distinguishes three major steps: perception, comprehension and projection; where the first of them is related with monitoring the environment and the discovery of initial facts, comprehension aims on the inference of knowledge and hence generating new facts from the observations, and projection is related with the prediction of the environment status. Note that the conventional model introduces feedback between data processing stages, in this way allowing learning and improving decision-making. As discussed in [CSD15], the Endsley model proven effectiveness in complex and dynamic scenarios where the diagnosis of incidents highly depends on their context. Throughout the bibliography it has been successfully combined with risk management models [FB14] and adapted to networking environments, consequently coining the term NSSA (Network Security Situational Awareness) [LM15]. Its adaptation to 5G started with project like 5G-ENSURE [R5G] which were mainly inspired in the risk assessment and management approaches.

More recently, SELFNET [P5S] adopted the situational awareness paradigm based in the research of Barona et al. [BLVCMV+17], which described a framework for hybridize the Endsley model, information security risk management and self-organizing networking. The perception stage of SELFNET was described in depth in [CV17], and a first approach toward orchestrating the activities related with comprehension and projection was published in [LVV17]. Situational awareness is in fact a key research topic related with information security in 5G networks as stated in the release of the first 5G PPP Phase 1 Security Landscape [BWSMea17], where the cognitive approximation to understand the network environment is tackled by a cognitive approach driven by contextual analysis.

The research conducted in [BLVCMV+17] proposed an approximation towards the implementation of the situational awareness stages for conducting autonomic incident management strategies by taking the SELFNET model as a reference architecture (Figure 3.2). As explained in the previous section; perception, comprehension and projection are attained by the monitoring, aggregation/correlation and analysis components of the SELFNET architecture.

Forthcomings chapters of this thesis delve into the procedures, methods and technology enablers for conducting reasoning processes oriented to achieve contextual analysis, thus emphasizing the projection stage as the main topic of this research. Thereby, to put in context the acquaintance of knowledge from the monitored network when dealing with incident management, two kinds of threats are studied in the following sections. They will ground the definition of the use cases to be studied.

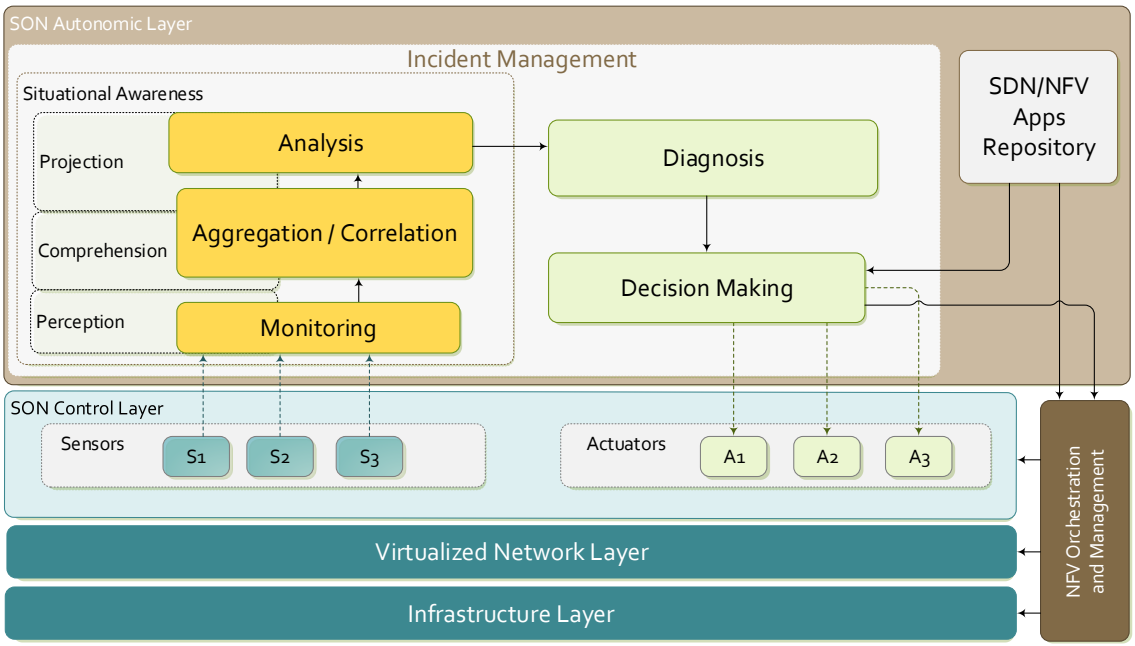


Figure 3.2: A Situational awareness approach for incident management in 5G.

3.3.1 Distributed Denial of Service Attacks

Nowadays there are different procedures that intentionally may lead to deplete the resources of a network element, thereby denying its service. The performed research focuses on those based on flooding the victim with malicious requests [ZJT13], which typically have been categorized in high-rate and low-rate DDoS attacks [WCXJ13]. This taxonomy considered as classification criterion their request frequency. According to this classification, the first family of threats gathers the techniques grounded in timely injecting a large volume of traffic/requests. On the other hand, low-rate DDoS intend to go unnoticed over the security measures by adopting incremental or activation/deactivation request injection patterns, that usually are less visible than conventional flooding-based intrusion attempts [BBK15]. Furthermore, the attacker may take advantage of reflection [XWY+17] and/or amplification [MSK17] to magnify the impact of the intrusion. A clear example of this is observed in the Link Flooding Attacks (LFA) [WLJW16], where low rate request flows from regions with high traffic density are reused aiming on overflowing the computing capacity of intermediate network elements, thus resembling legitimate traffic and making the threat difficult to be discovered. Flooding-based denial of service was originally achieved from a single point of the network, which was commonly referred as conventional Denial of Service (DoS) attacks. But the trend of the current devices towards gaining computing capacity, as well as the advances in cloud computing, and the tendency to implement self-scaling and load balancing mechanisms, entailed that nowadays the attacker requires a large number of devices (end-points) with traffic injection capabilities to reach their malicious purpose, this situation being typified as DDoS. Because of this, intruders usually resort to botnets [MDML17] for acquiring offensive power. Botnets are increasingly extensive and adapt to the emergent network scenarios [AAB+17]. In

addition, they have evolved towards robustness and evasion of mitigation techniques [VZF17], which make them a dangerous shuttle of denial of service attempts. However, given the great bibliography related to the botnet problem, this thesis does not deeply cover them, thus suggesting publications like [HBK15] for reviewing their most relevant features.

Most of the efforts of the research community dedicated to the defense against DDoS assumed the aforementioned circumstances. But given the complexity of the problem to be solved, the proposals are often divided into four different challenges [VZF17]: prevention, detection, mitigation and identification of sources. DDoS prevention focuses on avoiding the attack from reaching the victim, hence covering measures that range from applying filtering policies to traffic redistribution. Note that unlike mitigation, the prior identification of the intrusion with this purpose is not required. According to the bibliography, prevention approaches typically consider univocal features of the legitimate traffic [LLZZ13], Turing tests [WMLW18], security protocols [KVF+12] or reputation-based systems [WCC+17].

The research focused on DDoS detection requires studying features of the intrusion itself, either through the analysis of the traffic involved and/or the study of network-level behaviors [ZJT13]. To this end, different analytic techniques were adopted, among them hidden Markov model [HVV17], artificial neural networks [SOR16], entropy-based analysis [BBK15] support vector machines [AYON17] or decision trees, the latter discussing the efficiency of different machine learning methods. As highlighted in [YYGL16], within this group, the source-side DDoS detection played a minority role in the bibliography. It was classically addressed by validating the destination hosts of the outgoing traffic [SF00], or by recognizing discordant traffic patterns at flow-level [ZJT13]. For example, D-WARD [MPR03] proposed the construction of models that represented the normal usage of the traffic flowing through the protected system. It was based on classifying metrics extracted from traffic flows, from which it was possible to distinguish discordant behaviors. Another interesting contribution is illustrated in [GP01], where the proportionality between outgoing and incoming traffic is studied. With the advent of the SDN technologies, this detection paradigm [YYGL16] has been revised, leading to analyze the flow-tables inherent to the OpenFlow protocol. In this way DDoS attacks originated in groups of compromised end-points can be detected [MKK11], which usually are IoT or end-user devices [JW13]. But this requires conducting monitorization and feature extraction processes on data gathered in at least small/medium network regions.

Once the intrusion is recognized, the mitigation measures act. They involve the reactive and proactive deployment of some of the aforementioned prevention techniques, as well as the reinforcement of the network perimeters that displayed a higher risk level, which would lead to define quarantine regions [MVSOGV18]. They can also assume the instantiation of alternative countermeasures usually related to the active security model, among them deployment of honeypots and decoys [WDMS17], or the redirection of malicious traffic to sinkhole servers [JCG+17]. Within this security paradigm are framed the main techniques for identifying the source of the threats. Their principal objective is to discover the intruder, for which the adoption of traceback measures, highlighting among them packet

marking techniques, is frequent. In [Int13] some of the most popular source identification approaches are reviewed in detail. Note that because they often directly depend on the network topology [KBP14], and the fact that in real scenarios they suffer restrictions imposed by the different data protection policies [Den14b], their goal frequently tends to be simplified as to get as close as possible to the intruder, in this way allowing to extend the range of action of the instantiated security actuators.

DDoS has been extensively studied in the research literature, hence remaining as a hot security topic. It has benefited from the advances of modern technologies, thus sophisticating intrusion and evasion techniques and also gaining complexity in emerging scenarios. This is a critical aspect to be considered when defining defensive strategies against them.

3.3.2 Economic Denial of Sustainability Attacks

At November 2008, Hoff [Hof08] [Hof09] firstly hypothesized about the presence of a novel strain of the denial of service threat, which was termed Economic Denial of Sustainability, abbreviated EDoS. It described a specific family of attacks against the different cloud computing platforms, where the intruder aimed on increasing the economic costs derived from both maintenance and provision of the services offered, hence making their support less viable, even achieving denial. Interested in this publication, R. Cohen [Coh09] extended the EDoS definition by highlighting the important role played by exploiting the self-scaling mechanisms considered by each provider. That is, if the attacker succeeds in forcing the users to hire additional computing resources by exploiting the self-escalation policies of the provider, clients will have to pay more, which may lead them to change to a more competitive service supplier. This implies that the service offered ceases to be profitable, and that hence the EDoS attack achieved its main purpose.

Although it is a novel concept, inherent to the emergent technologies, EDoS rapidly drawn the attention of the research community and organizations for information security, according to them becoming a DDoS variant framed in the categories Reduction of Quality (RoQ) [BBBS17] and Fraudulent Resource Consumption (FRC) threats [SGSC16]. They stated that EDoS typically attempt to exploit the "pay-as-you-go" service model offered by most of the cloud computing providers [SMR14] [SGS+17], which leads to its adaptation to different metrics, and self-scaling policies or mechanism [BBBS17]. The following describes its main characteristics, impact and the defensive strategies raised by the research community.

3.3.2.1 Characteristics and impact

In general terms, EDoS attacks pose similarities with conventional DDoS threats, especially those based on flooding [ZJT13]. However, EDoS pose a significantly different problem that requires a separate solution: assuming the original definition of Hoff [Hof08][Hof09], EDoS focuses on forcing the increase of the economic cost of a cloud computing service instead of directly preventing its provision, as occurs in their predecessors. Therefore, for the sake of effectiveness the number of connections and

requests involved in their execution may resemble the legitimate activities. Hence network metrics traditionally studied when detecting flooding-based threats (e.g. number of requests, number of sessions, total amount of payload, bandwidth consumption, etc.) display distributions similar to those gathered at normal traffic. Therefore, the attacker usually exploits vulnerabilities at application layer with the purpose of extend the computational cost of resolving the received requests [SGSC16]. In this last characteristic lies the greatest difference between EDoS attacks and massive accumulations of legitimate requests, the second commonly referred as flash crowds [YZJ+12], in which the processing cost is similar to those of the requests monitored before the agglomeration.

The computational cost of attending the received requests can be exploited to induce EDoS in several ways, for example by requesting large files or costly queries [BS15], HTTP-requests on XML files [VS12], or taking advantage of alternative Application layer vulnerabilities related with web services [ZJW+14] [SSK17][SC17]. G. Sonami et al. [SGSC16] studied in-depth the impact of these threats, concluding that it varies depending on the affected party. In the case of the customers, EDoS directly involves economic losses, which can lead them to the election of more economical suppliers. This indirectly causes the providers to lose customers, and therefore their profit is reduced. As discussed in [SC17], the increase in the computation requirements for addressing the malicious requests also implies a decrease in the quality of service offered, which is a consequence of the need for among others, more infrastructure, deployment of additional network virtualization functions (NFV), multi-tenancy capabilities, etc. It was defined by Bremler-Barr et al. [BBBS17] as collateral damage, publication in which they demonstrated that by launching an easy low-rate EDoS attack, it was possible to prompt additional causalities directly related with enforce auto-scaling and the waste of resources it involves.

3.3.2.2 Defense against EDoS

Despite relevance, the bibliography does not include a large collection of publications focused on the defense against EDoS threats. The studies that address this problem usually assume metrics at network-level, usually confusing features for EDoS identification with those that typically detect flooding-based DDoS attacks. The nature of EDoS threats poses instead resemblance with normal network traffic behaviors, hence requiring different defensive approaches to find particular discordances both at network and application level. In [BS15][BM15][BSB16] some of the most relevant proposals are collected and discussed. With the purpose of facilitate their understanding, they are classified according to their scope, as traditionally organized in the research related with the defense against DDoS [IT11]: detection, prevention/mitigation, and identification of sources.

- *Detection.* The approaches aimed on detection share the main goal of identify the threats. Therefore, they are often the triggering situation prior to the deployment of mitigation capabilities or the identification of the attack sources. In the bibliography there are two types of proposals assuming as classification criteria the scope of the metrics to be analyzed: those extracted at local, and network monitoring environments. Publications focused on local metrics traditionally model the

consumption of resources and study the self-scaling processes [IT11]. Network-based approaches analyze information provided by the packet headers [AHSS12][JSTD16], and the browsing habits of the normal clients [KNB13][ITJ12]. Although few investigations have focused on local traits, they demonstrated greater efficacy, since they directly assume the definition of EDoS attacks originally posted by Hoff [Hof08][Hof09]. However, methods based on network traits analysis take advantage of the state of the art about flooding-based DDoS detection, which in many cases has led to confusion between both types of attacks.

- *Mitigation and Prevention.* Once the threat has been succesfully identified, the proposals for mitigation act. They mainly focus on increasing the protected system restriction level through the deployment of more complex access control techniques, usually Turing tests based on image recognition [KSK+12] [AAB13] and resolution of cryptographic puzzles [MARH13][KN09][KSK+12]. It is important to highlight that most of the publications gathered in the state of the art addressed the mitigation problem by the aforementioned classical solution for cloud computing security incidents. On the other hand, publications for preventing EDoS threats aimed on modeling and optimizing costs related with processing the malicious requests [YTGW14]. Unlike mitigation approaches, they do not require the prior detection of the threat. Notwithstanding, most of the proposals categorized in mitigation could be deployed as prevention measures.

- *Identification of sources.*Finally, the research that aims on identifying the origin of the attacks attempt to discover the attacker itself. Given the difficulty that this entails, and the fact that it is often not possible due to the restrictions of Internet providers, as well as intermediate elements of the backbone, privacy and data protection policies, etc. from the practical point of view, identifying the origin is often simplified at reach as close as possible to the attacker. Most of the classic techniques in the state of the art of the defense against conventional DDoS serve for this purpose [ITJ12], among them those based on the analysis of error messages [AR14], deployment of honeypots [WDMS17] or packet marking [YBV15], in all of them playing the network topology an essential role [JL14].

Unlike DDoS, EDoS has recently drawn the attention of the research community. Thus, it opens the possibility to evaluate its impact in modern and complex network scenarios where virtualization plays a critical role. The literature has mostly framed the study of EDoS threats in cloud deployments, however, and as discussed in Sections 2.4 and 3.2, the immersion of virtualization technologies in 5G infrastructures arises new security concerns to be addressed. Therefore, later chapters of this thesis delve into the study of EDoS threats not only in cloud environments, but also in the contexts of self-organizing network scenarios.

3.3.3 Towards Crypto-ransomware Mitigation in 5G: A Self-organizing Approach

Throughout last decade, crypto-ransomware evolved from a family of malicious software with scarce repercussion in the research community, to a sophisticated and highly effective intrusion method positioned in the spotlight of the main organizations for cyberdefense. Its modus operandi is characterized by fetching the assets to be blocked, their encryption, and triggering an extortion process that leads the victim to pay for the key that allows their recovery. In this section, a novel defensive approach based on the Self-Organizing Network paradigm and the emergent communication technologies proposed in [SMMVGV18a] is examined. The proposal exemplifies the smart management of network-based countermeasures against crypto-ransomware without human supervision, as well as the adaptation of the defensive deployment to the state of the monitored environment in a context of new generation networks. This is possible by instantiating/removing sensors and actuators according to their effectiveness and the risk level of the region they operate; and by establishing regions of quarantine for stronger monitoring and actuation. They enhance the orchestration of smart defensive deployments that adapt to the status of the monitoring environment and facilitate the adoption of previously defined risk management policies. In this way it is possible to efficiently coordinate the efforts of sensors and actuators distributed throughout the protected environment without supervision by human operators, resulting in greater protection with increased viability. Recently, and in parallel with the development of local-level solutions, the research community studied the impact of the crypto-ransomware on communication processes. This prompted the publication of the first proposals based on analyzing network features in emerging scenarios, as is the case of [Zah17] at Internet of Things (IoT) or [Lee17] at Cloud Computing. Hence, such approaches raises awareness on the importance of deploying more advanced defensive schememes adapted to emerging networks and, in particular, to 5G architectures.

According to the standardized framework for SON 3GPP networks, a closed-loop approach allows a reduction of the operational costs, hence streamlining decision-making and counteracting. As highlighted in [HSS12b], even the simplest self-organized networks govern their behavior based on information monitored by sensors. Therefore, the proposed defense against crypto-ransomware has as starting point data collected by sensors scattered throughout the protected environment. In order to generate high level metrics and identify traits of suspicious behaviors, the monitored information is aggregated and analyzed at a different data processing plane. Note that with the purpose of minimizing the impact of the sensor at the end-points both aggregation and analysis tasks are launch on dedicated servers. From the acquired situational knowledge it is decided when to deploy or remove the additional security measures, which are executed by actuators defined as agents in charge of closing the loop by enforcing security policies.

The algorithm that manages the situational awareness and dictates the immune responses per network region is illustrated in Figure 3.3. There three intelligent loops are described, each of them having a different purpose (see Table 3.1). The first closed

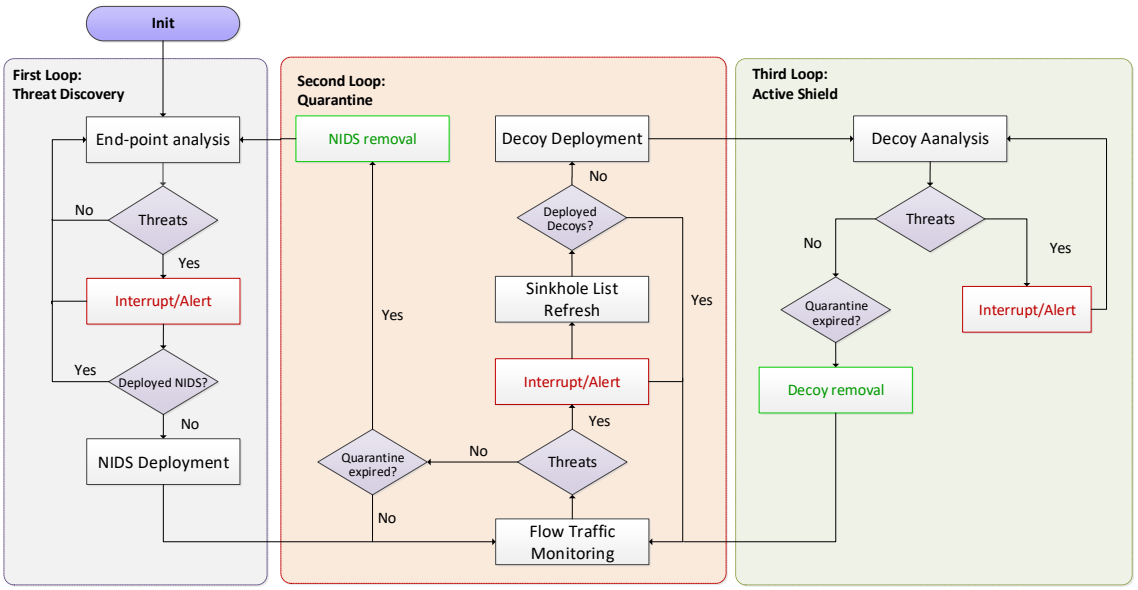


Figure 3.3: Self-organization per defended region as flowchart

loop is termed Threat Discovery and focuses on recognizing discordant behaviors typical of cryptomalware at the end-points. This process runs continuously and serves as triggering for the most basic network-level actuations, which entail the deployment of additional monitoring elements at regions suspected of having been compromised. During the second closed loop, referred as Quarantine, they perform as sensors that aim to manage the passive security countermeasures deployed at the affected regions (areas in quarantine), hence deciding how long they should remain operational, their configuration, and if it is required to increase their restriction level. Therefore, each iteration of the second loop has three possible consequences: leaving the defensive deployment as it stands, recalling actuators (i.e. dismantle quarantined regions), or incorporating active security measures [Den14a] (which behave much more restrictive); in particular, the latter is accomplished by adding decoys and sink servers [Kea16]. If it is decided, the third closed loop is triggered, which is refereed as Active Shield. This loop is responsible for managing the active security countermeasures, hence orchestrating a second quarantine layer limited by their range of action. Its execution can lead to two possible consequences: maintaining the active defensive deployment or dismantling the second quarantine layer, which occurs when the level or risk significantly decreases.

The three closed-loops described so far introduced a novel self-organizing defensive approach which allows the smart coordination and calibration of countermeasures deployed as sensors and actuators. Such defensive scheme orchestrates their instantiation considering the acquired situational awareness of the protected environment and risk level. It has empowered by the adoption of emerging communication technologies inherent to the progress towards the development of new generation networks (5G). Therefore, this approximation discloses the applicability of knowledge-based autonomic management towards the mitigation of crypto-ransomware threats, whilst it opens interesting lines for future research.

Table 3.1: Closed-loops for crypto-ransomware defense

Feature	First loop	Second loop	Third loop
Name	Threat Discovery	Quarantine	Active Shield
Quarantine region triggering	First-level	Second-level	
Quarantine region management		First-level	Second-level
Sensors	HIDS	DPI	Decoys
	Sandboxes		Sinkholes
Actuations	Local process interruption	Custom handshake interruption	Traffic redirection
	Notification	Notification	Sinkhole configuration
	DPI and NFVs instantiation	Decoy NFVs instantiation	Data gathering
		Sinkhole configuration	Notification

3.4 Final remarks

Research initiatives are performed worldwide to reach the maturity level expected for 5G networks bearing in mind the timeline towards the year 2020, when it is expected to find the first release of this technology. The SELFNET project is aligned to that strategy tackling with the autonomic network management on self-organized environments. The accomplishment of this relies on the ability to acquire knowledge to understand the network context in order to perform efficient decision-making processes. The Situational Awareness model proposed by Endsley contributes to this purpose by providing a contextual-based analysis model on which the stage of projection aims to infer the current and future status of the monitored network. This approximation gains relevance in the incident management lifecycle to attain cognitive-based defensive approaches. To accomplish the research objectives of this thesis, the state of the art of two network threats were reviewed for immersing into knowledge-based detection strategies later in Chapters 7 to 9, where the proposed use case scenarios and countermeasures suited for self-organizing networks are studied in detail.

Chapter 4

Prediction Algorithms and Adaptive Thresholding

This chapter introduces the prediction algorithms and adaptive thresholding methods applied throughout this research, which provide forecasting capabilities in the knowledge acquisition process, as part of the autonomic network management. The contents of this chapter are organized as follows: Section 4.1 describes the prediction landscape on networking. Section 4.2 describes well-known forecasting algorithms for univariate analysis. Section 4.3 introduces the adaptive thresholding approach considered to estimate a prediction interval for analysis purposes. Finally, Section 4.4 remarks the conclusions of this chapter.

4.1 Network prediction

Anticipating the occurrence of network incidents or events grants the network the capability to react proactively when dealing with network threats that might produce a degradation of the service quality or, in the worst case, the unavailability of the service. Such approach can be conducted by predicting the evolution of monitored network metrics over time. When sampled, measured or collected on periodic time intervals; those metrics can be represented as time series, which leads to the application of prediction algorithms for estimating their behaviour. Given the heterogeneity of the monitored metrics, not all the algorithms might be suitable for performing prediction since they take into account different time series attributes such as the trend, seasonality, variance, among others. Hence, some prediction methods perform better when assessed on a given time series.

5G networks envision heterogeneous scenarios where context-based prediction plays an essential role in several domains such as geographic, network link, traffic mesurements, social-type, among other network contexts. The deployment of proactive actions enables an autonomic approach where the network can take benefit of future conditions for accomplishing the Key Performance Indicators (KPIs) established by 5G [BCH+17].

Thereby, the prediction algorithms introduced in this chapter are intended to drive the inference of knowledge related with forecasted data in the detection of network scenarios that can led to the inference of discordant behaviours. Three main categories of algorithms

are identified to this end: moving average, smoothing, and autorregresive models.

On the other hand; rather than being exact, predicted metrics need a confidence interval on which its precision can be assessed. To this end, an approach for building adaptive thresholds taken the forecasted estimation as a baseline is mandatory. As a result, prediction interval should be estimated for assessing the behaviour of the monitored metrics, which is introduced at the end of this chapter.

4.2 Prediction Algorithms

The most relevant prediction methods for univariate analysis on time series are described in this section, which are summarized below.

4.2.1 Cumulative Moving Average

Let a sequence of serialized data $x_1, x_2, \ldots, x_n$, the purpose of the Cumulative Moving Average (CMA) [Men15] is to calculate the average of the data stream CMA_i, $0 < i \leq n$ from the first observation until the element x_i. This matches with the following recursive expression:

$$CMA_n = \frac{\sum_{i=0}^{n} x_i}{n} \tag{4.1}$$

where the CMA at $n+1$ is expressed as follows:

$$CMA_{n+1} = \frac{x_{n+1} + n \times CMA_n}{n+1} \tag{4.2}$$

deduced by assuming $x_1 + \ldots + x_n = n \times CMA_n$.

This approach is also known as running average or long running average. Usually considered as a smoothing method for time series that equally takes into account all the registered observations, the CMA provides a good baseline to infer predictions. Figure 4.1 illustrates an example of prediction by CMA estimation on a randomly generated time series.

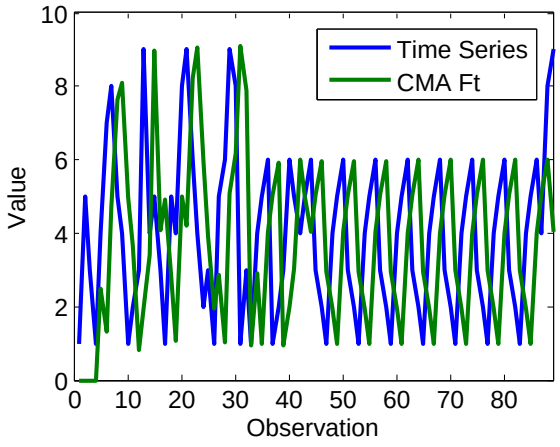


Figure 4.1: Example of prediction by CMA

4.2.2 Simple Moving Average

The Simple Moving Average (SMA) is another smoothing strategy based on the mean of the last n observation within a time series [JBMS99]. It can be expressed as a constrained variation of the CMA, where m is the length of the subsequence to be taken into account. The SMA is formulated as follows:

$$SMA = \frac{P_m + P_{m-1} + \ldots + P_{m-(n-1)}}{n} = \frac{1}{n}\sum_{i=0}^{n-1} P_{m-i} \tag{4.3}$$

and the next SMA is forecasted as expressed below:

$$SMA_{t+1} = SMA_t + \frac{P_m}{n} - \frac{P_{m-n}}{n} \tag{4.4}$$

As is the case of CMA, throw the addition of the previous registered absolute errors it is possible to infer the future values of the analyzed time series. The higher is m, the greater is the similarity with the CMA. Hence the smoothing is more relevant, but also the time consumption of the algorithm. An example of predictions form $m = 4$ on randomly generated time series is illustrated in Figure 4.2.

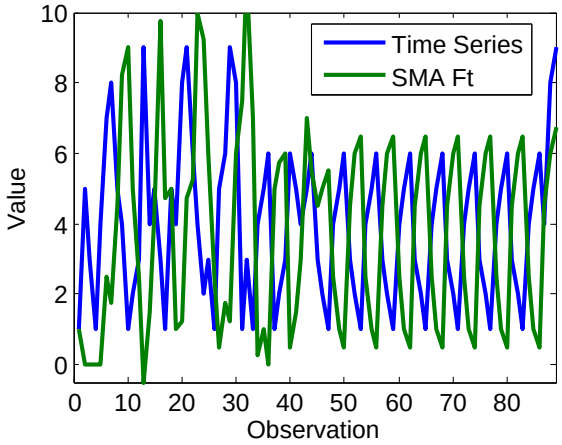


Figure 4.2: Example of prediction by SMA $m = 4$

4.2.3 Double Moving Average

The Double Moving Average (DMA), was introduced by Mullony [KW08][Mul94a][JBMS99] with the purpose of mitigate the time consumption of the traditional moving average compositions. In the previous bibliography, and especially at financial domain, it was a common practice to repeat the SMA calculation in order to highlight specific indicators related with the analysis of stocks and commodities. Given the following SMA refereed as M_t for a time series of observations:

$$M_t = \frac{Y_t + Y_{t-1} + \ldots + P_{t-(n+1)}}{n} \tag{4.5}$$

and the following SMA $M_t^{'}$ contructed from its smoothing:

$$M_t^{'} = \frac{M_t + M_{t-1} + \ldots + M_{t-(n+1)}}{n} \tag{4.6}$$

DEMA is formulated as follows:

$$DMA_t = 2M_t - M_t^{'} \tag{4.7}$$

The prediction of future observations is based on the parameter b_t expressed as:

$$b_t = \frac{2}{n-1}\left(M_t - M_t^{'}\right) \tag{4.8}$$

which allows inferring the $\hat{Y}$ observation in $t+p$ as follows:

$$\hat{Y} = DMA_t + b_t p \tag{4.9}$$

So as is the case of SMA, DMA inherits the adjustment parameter m for defining the size of the considered observation window. Figure 4.3 illustrated the prediction calculated for $m = 4$.

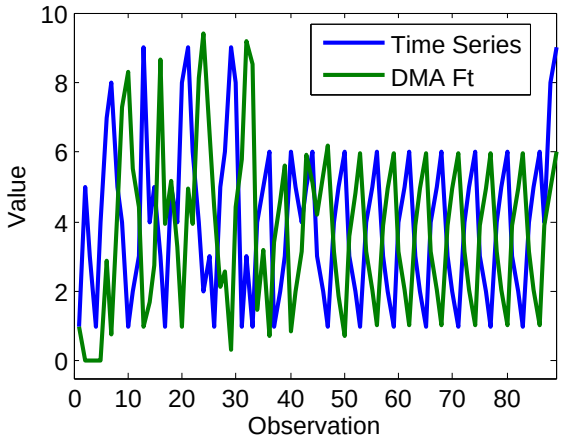


Figure 4.3: Example of DMA prediction with m=4

4.2.4 Weighted Moving Average

In contrast with the previous moving averages, the Weighted Moving Average (WMA) considers different multiplicatively weights to the observed data in different positions of the time series [FL14]. This allows attaching more importance to recent events, hence providing more quickly reactions to the recent changes. WMA is widely implemented for calculate different indicators, such as the trend direction, support and resistance areas at financial domain and accommodate forecasting. Let the $x_1, x_2, \ldots, x_n$ sequence, it is formulated as follows:

$$WMA_t = \frac{w_t x_t + w_{t-1} x_{t-1} + \ldots + w_{t-(n+1)} x_{t-(n+1)}}{n + (n-1) + \ldots + 2 + 1} = \frac{\sum_{t=1}^{n} w_t x_t}{\sum_{t=1}^{n} w_t} \tag{4.10}$$

where $w_i, 1 \geq i \leq n$ is the weight for the observation at i. The current implementation

of WMA_t assumes the classical $w_i = i$ weighting definition. As is the case of SMA, predictions are obtained through addition of the previous registered absolute error between the WMA and its corresponding observation, to the last observation (see Figure 4.4).

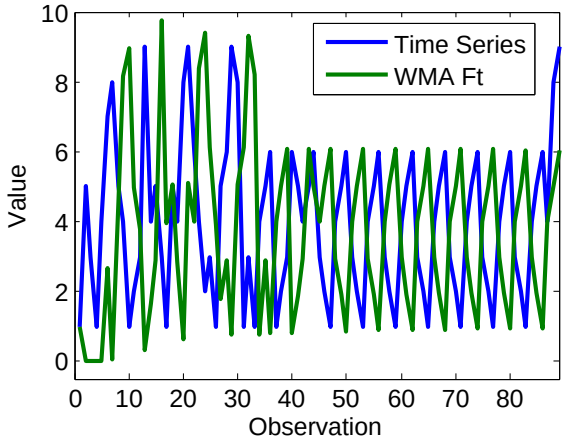


Figure 4.4: Example of prediction with WMA

4.2.5 Simple Exponential Moving Average

The Simple Exponential Smoothing (EMA) method [ASMW15], also known as exponentially weighted moving average (EWMA), is another fast response strategy that providers grater novelty discovery at the expense of being more prone to abrupt fluctuations (i.e. false signals). Unlike WMA, the EMA weighting factors decrease exponentially, so it can be considered a particular case of WMA which satisfied such feature. It is usually formulated as the following recursive expression:

$$EMA_1 = x_1 \tag{4.11}$$

$$EMA_t = \alpha x_t + (1 - \alpha)\, EMA_{t-1} \tag{4.12}$$

where $\alpha, 0 <= \alpha <= 1$ is the adjustment parameter of the degree of the weighted decrease. The higher α the most relevant are the new observations.

As it is illustrated in Figure 4.5, $\alpha = 0.4$ provides a smoothed base for infer the next observations when performing prediction with EMA.

4.2.6 Double Exponential Moving Average

It is typical that the application of EMA in the financial domain results in the requirement of calculating several variations of EMA for different time periods and degrees of the weighted decrease, and they contrast. This used to be a very expensive approach in terms of computational resources, hence motivating the publication of similar methods. Among them it is important to highlight the Double Exponential Moving Average (DEMA) proposed by Mulloy. Similarly to DMA, DEMA is calculated for a time series of observations as follows:

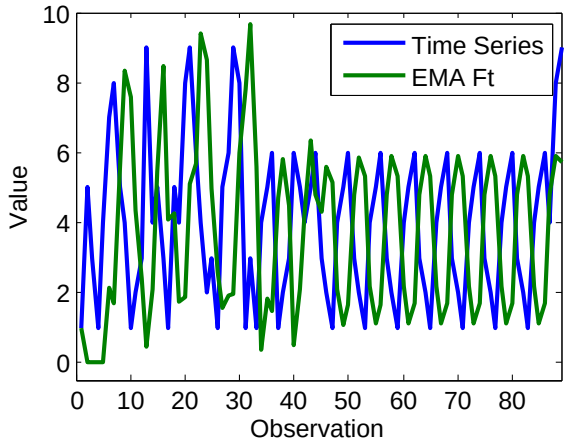


Figure 4.5: Example of prediction with EMA for $\alpha = 0.4$

$$EMA_1 = x_1 \tag{4.13}$$

$$EMA_t = \alpha x_t + (1 - \alpha)\, EMA_{t-1} \tag{4.14}$$

and the following $EMA_t^{'}$ constructed from its smoothing:

$$EMA_1^{'} = x_1 \tag{4.15}$$

$$EMA_t^{'} = \alpha x_t + (1 - \alpha)\, EMA_{t-1} \tag{4.16}$$

DEMA is formulated as follows:

$$DEMA_t = 2EMA_t - EMA_t^{'} \tag{4.17}$$

The prediction of future observations is based on the parameter b_t expressed as:

$$b_t = \frac{2}{n-1}\left(EMA_t - EMA_t^{'}\right) \tag{4.18}$$

which allows inferring the $\hat{Y}$ observation in $t + p$ as follows:

$$Y = DEMA_t + b_t p \tag{4.19}$$

An illustrative example of prediction with DEMA for $\alpha = 0.4$ is illustrated in Figure 4.6.

4.2.7 Triple Exponential Moving Average

As an alternative to DEMA P.G. Mulloy proposed the Triple Exponential Moving Average (TEMA) [Mul94a]. It provides an additional smoothing level, which is calculated from the EMA:

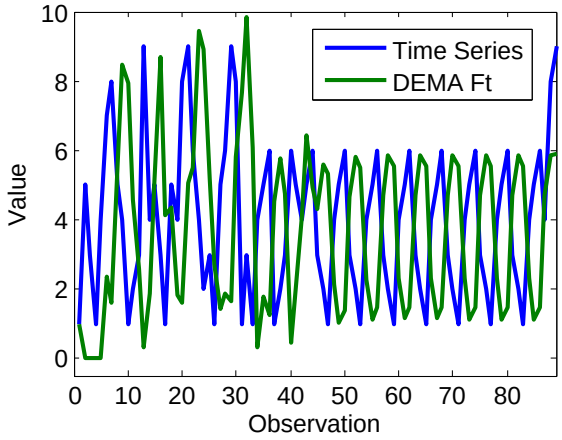


Figure 4.6: Example of prediction with DEMA for $\alpha = 0.4$

$$EMA_1 = x_1 \tag{4.20}$$

$$EMA_t = \alpha x_t + (1 - \alpha)\, EMA_{t-1} \tag{4.21}$$

the following $EMA_t^{'}$ contructed from its smoothing:

$$EMA_1^{'} = x_1 \tag{4.22}$$

$$EMA_t^{'} = \alpha x_t + (1 - \alpha)\, EMA_{t-1} \tag{4.23}$$

and the $EMA_t^{''}$ that considers the previous baselines:

$$EMA_1^{''} = x_1 \tag{4.24}$$

$$EMA_t^{''} = \alpha x_t + (1 - \alpha)\, EMA_{t-1}^{'} \tag{4.25}$$

so TEMA is summarized as follows:

$$TEMA_t = 3EMA_t - 3EMA_t^{'} + EMA_t^{''} \tag{4.26}$$

Figure 4.7 illustrates an example of prediction with TEMA calculated when $\alpha = 0.2$.

4.2.8 Simple Exponential Smoothing

The Simple Exponential Smoothing (SES) was introduced by R.G. Brown [Bro57] and extended by C.C. Holt [Hol04] as an extension of an analytical approach attributed to Poisson. It is a variation of EMA suitable for predict observations on time series with no trend or seasonal pattern, represented by the following recursive expression:

$$S_t = \alpha y_{t-1} + (1 - \alpha)\, S_{t-1} \tag{4.27}$$

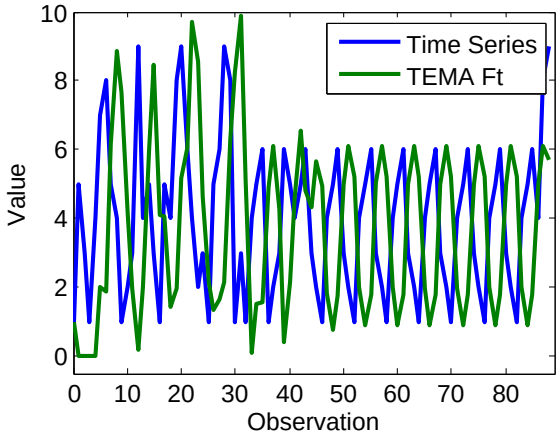


Figure 4.7: Example of prediction with TEMA

where $0 < \alpha < 1, t >= 3, y_i$ is the observation at i, and α is the smoothing constant. There are several approaches to the problem of fix the base case of this expression. Hence, the prediction approach assumes the classical $S_2 = y_1$, postponing the exploration of alternative strategies for future work. On the other hand, as frequently observed in the bibliography, the adjustment of the parameter α is obtained by calculating the values minimizing the function Sum of the Squared Errors of prediction (SSE) at the initialization observations, defined as:

$$SSE\left(\alpha,\right) = \sum_{t=1}^{N} \left(H_\alpha\left(X\right)_t - H_\alpha\left(X\right)_{t_{|t-1|}}\right)^2 \tag{4.28}$$

On this basis, the forecasted values are calculated as follows:

$$S_{t+1} = \alpha y_t + \left(1-\alpha\right) S_t \tag{4.29}$$

which is also expressed as:

$$S_{t+p} = S_t + \alpha\epsilon_t \tag{4.30}$$

where ϵ_t is the prediction error observed at t. Figure 4.8 illustrates an example of prediction with SES. Given that the adjustment parameter is auto-fitted, the forecast demonstrated a very accurate behavior.

4.2.9 Double Exponential Smoothing

By definition, SES may not operate effectively when there is a trend in the analyzed time series. In order to mitigate this drawback, the Double Exponential Smoothing (DES) algorithm was proposed [GJ06]. It introduces an additional constant γ related with the degree of trend, and a second equation taking it into account. The new recursive ecuations are detailed below:

$$S_t = \alpha y_{t-1} + \left(1-\alpha\right)\left(S_{t-1} + b_{t-1}\right) \tag{4.31}$$

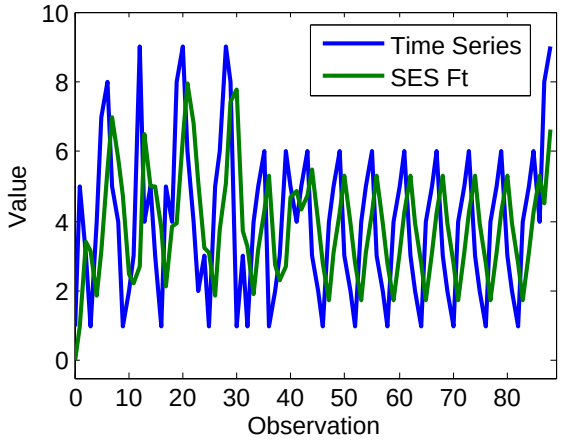


Figure 4.8: Example of prediction with SES

$$b_t = \gamma \left(S_t - S_{t-1} \right) + \left(1 - \gamma \right) b_{t-1} \tag{4.32}$$

where $0 <= \alpha >= 1, 0 <= \gamma <= 1$. As is frequent in the bibliography, the case bases are the initializations $S_t = y_1$ and b_1 may be:

$$b_1 = y_2 - y_1 \tag{4.33}$$

$$b_1 = \frac{1}{3} \left[(y_2 - y_1) + (y_3 - y_2) + (y_4 - y_3) \right] \tag{4.34}$$

$$b_1 = \frac{y_n - y_1}{n - 1} \tag{4.35}$$

Predictions based on these methods are then calculated as follows:

$$y_{t+1} = S_t + b_t \tag{4.36}$$

$$y_{t+m} = S_t + m b_t \tag{4.37}$$

An example of their implementation is illustrated in Figure 4.9.

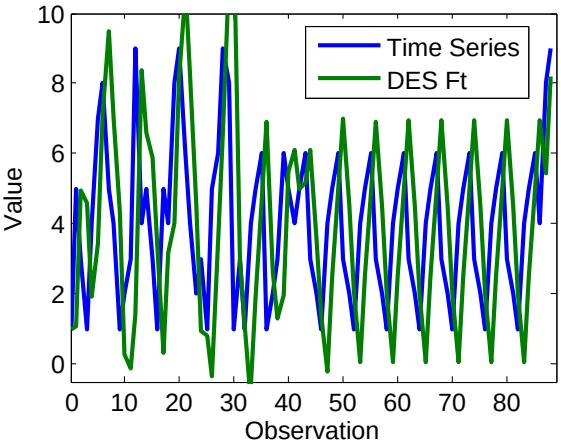


Figure 4.9: Example of prediction with DES

4.2.10 Triple Exponential Smoothing

Unlike DES, the Triple Exponential Smoothing (TES) algorithm takes into account the seasonal changes of time series [Win60]. Therefore, it incorporates a new adjustment value β related with the seasonal degree and an addition recursive expression, so it is calculated by:

$$b_t = \alpha\left(y_t - S_{t-N}\right) + (1-\alpha)\left(b_{t-1} + T_{t-1}\right) \tag{4.38}$$

$$T_t = \beta\left(b_t - b_{t-1}\right) + (1-\beta)\, T_{t-1} \tag{4.39}$$

$$S_t = \gamma\left(y_{t_t} - b_t\right) + (1-\gamma)\, b_{t-N} \tag{4.40}$$

where b_t is the base estimation at t, the estimation of the trend is T_t and the estimation of the seasonal factor is S_t. The parameters α, β, γ fall in the range $0 < \alpha, \beta, \gamma < 1$, and facilitate the adjustment of the smoothing. The prediction prediction y_{t+m} is usually calculated by additive or multiplicative operations, being additive:

$$y_{t+m} = mb_t + T_{t-m} + S_t \tag{4.41}$$

and multiplicative:

$$y_{t+m} = \left(S_t + mb_t\right) T_{t-m} \tag{4.42}$$

The first one is recommended for analyzing time series with significant trend and additive seasonal component, and the second is best suited for data with multiplicative seasonal component. Another important aspect to keep in mind is the initialization method of b_0, T_0, S_0 estimators. It is assumed that when no trend or seasonality is expected on the time series, the initialization of estimators based on the latest observations is preferable over the use of global measures. The implemented method is described in [MWH97], which has proven to behave particularly well in similar use cases. Namely, the last twenty-four observations are considered. The calculations performed are the following:

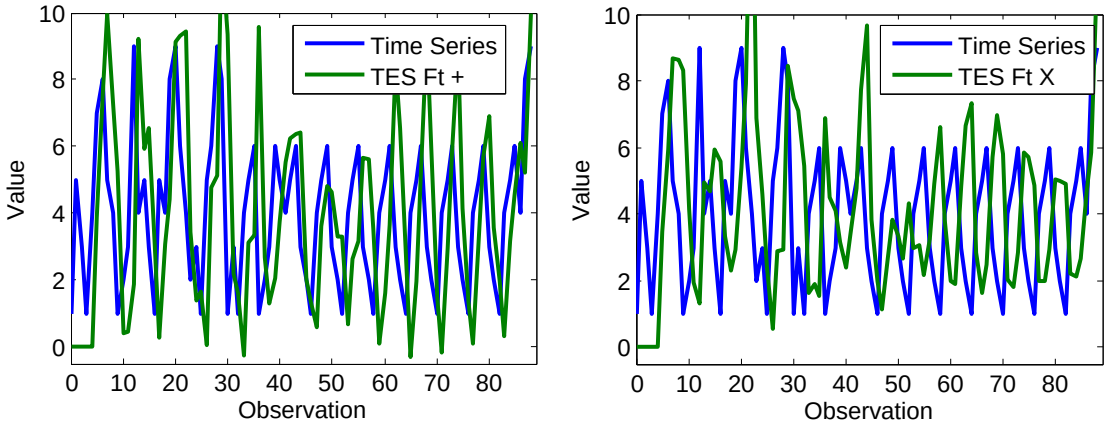
$$b_0 = M_1 \tag{4.43}$$

$$T_0 = \frac{M_2 - M_1}{12} \tag{4.44}$$

$$S_{t-12} = \frac{p_t}{M_1} \tag{4.45}$$

where M_1 summarizes the first twelve observations and M_2 the last dozen. The adjustment of parameters α, β, γ is obtained by calculating the values minimizing the sum of the squared errors of the prediction.

Figure 4.10(a) and Figure 4.10(b) illustrated examples of the results of forecasting with Holt-Winters.



(a) Additive (b) Multiplicative

Figure 4.10: Example of prediction with TES.

4.2.11 Autoregressive Models

The Autoregressive Model family, unlike the exponential smoothing approaches, is not based on decomposing the datum into factors. Instead its output variables depend linearly on the previous observations and on a stochastic term.

4.2.11.1 Classical Autoregressive Model (AR)

The classical AutoRegressive $AR(p)$ [Aka71] model is defined as:

$$Y_t = \mu + \phi_1 Y_{T-1} + \ldots + \phi_P Y_{T-P} + \epsilon_t = \mu + \sum_{i=1}^{N} \phi_i Y_{T-i} + \epsilon_t \tag{4.46}$$

where $epsilon_t$ is white noise (noise with mean cero), $phi_1 \ldots \phi_p$ are the parameters of the model, μ is a constant value, and p is the order (number of time lags) of the autoregression. The seasonal condition must be satisfied in the autoregressive part, and it is required for the adjustment of ARMA models.

4.2.11.2 Moving Averages Model (MA)

The Moving-Average $MA(q)$ [SD84] model is a different approach, where the output variable depends linearly on the current and various past observations, so learn of the past errors is possible. Note that q is the order of the moving-average model. MA instantiations are defined by the following expression:

$$Y_t = \mu\phi_1 Y_{T-1} + \ldots + \phi_P Y_{T-P} + a_t - \theta_1 a_{T-1} - \ldots - \theta_p a_{T-q} \tag{4.47}$$

equivalent to:

$$(1 - \phi_1 B - \ldots - \phi_p B^p)\, Y_t = \mu + (1 - \phi_1 B - \ldots + \phi_p B^p)\, a_t \tag{4.48}$$

and summarized as:

$$\phi_p(B)\, Y_t = \mu + \Theta(B)\, a_t \tag{4.49}$$

Unlike AR, they are always seasonal, but must satisfy the invertibility condition.

4.2.11.3 Autoregressive Integrated Moving Average (ARIMA) Model

The combination of AR and MA lead to the definition of the Autoregressive Integrated Moving Average $ARIMA(p, d, q)$ [HT82] model, also known as Box-Jenkins models, which is a generalization of ARMA that overcome its inoperability with non-seasonal data, where d is the degree of differencing (the number of times the data have had past values subtracted in order to became seasonal). A classical ARIMA model is expressed as follows:

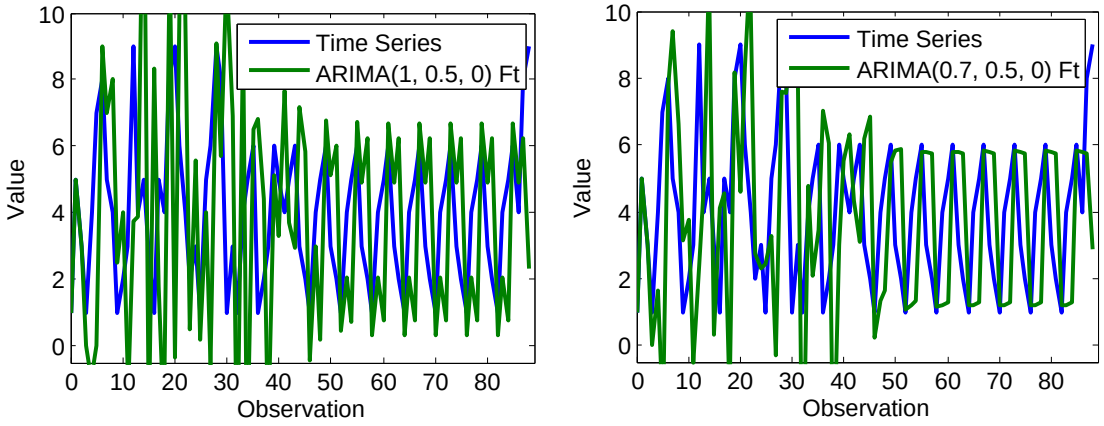
$$Y_{T-1} - a_1 Y_{T-1} - \ldots - a_{p'} Y_{T-p'} = \epsilon_t + \theta_1 \epsilon_{t-1} + \ldots + \theta_q \epsilon_{t-q} \tag{4.50}$$

where a_i are the parameters of the autoregressive part, θ_i are the parameters of the moving average part and ϵ_t is the white noise. The adjustment of p, d, q may be the ARIMA model equal to other forecasting models. For example ARIMA(1,1,0) is simply random walk, ARIMA(1,0,0) is AR, ARIMA(0,0,1) is MA, ARIMA(0,0,0) is white noise, ARIMA(0,1,1) is simple exponential smoothing, ARIMA(0,2,2) is double exponential smoothing, etc. The calibration of the best suited algorithms can be performed by several strategies, highlighting among them the Akaike Information Criterion (AIC) and the Gaussian approximation Criterion.

Predictions on ARIMA models are generated by a generalization of the autoregressive forecasting method where:

$$Y_t = \mu + \phi_1 Y_{T-1} + \ldots + \phi_P Y_{T-P} - \theta_1 \epsilon_{t-1} - \ldots - \theta_q \epsilon_{t-q} \tag{4.51}$$

Figure 4.11(a) and Figure 4.11(b) illustrate examples of ARIMA based forecast.



(a) ARIMA(1,0.5,0)

(b) ARIMA(0.7,0.5,0)

Figure 4.11: Example of prediction with ARIMA.

4.3 Adaptive Thresholding

For the evaluation of the prediction errors, two adaptive thresholds are constructed: an upper threshold Ath_{up} and a lower threshold Ath_{low}. They establish the Prediction Interval (PI) of the observation, which is defined in the same way as is usually performed in the bibliography [MWH97], hence assuming the following expressions:

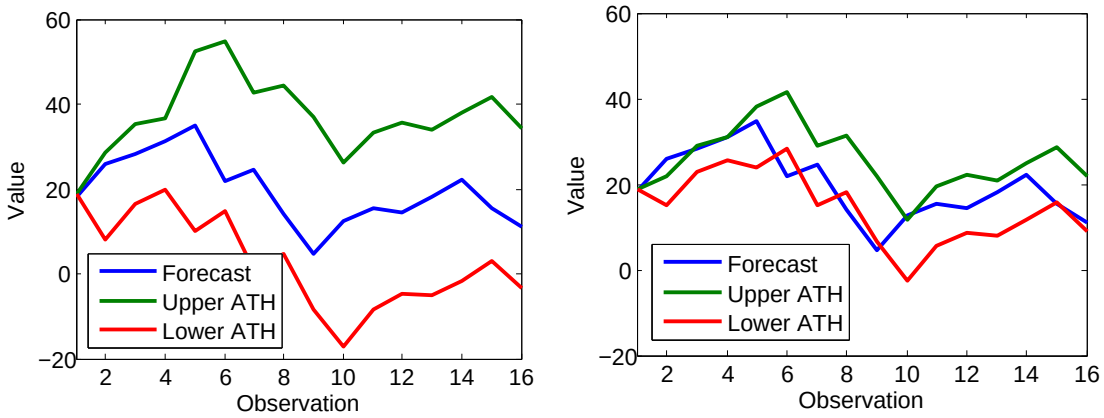
$$Ath_{up} = \hat{x}_{n+1} + K\sqrt{\sigma^2(E_t)} \tag{4.52}$$

$$Ath_{down} = \hat{x}_{n+1} - K\sqrt{\sigma^2(E_t)} \tag{4.53}$$

where $\hat{x}_{n+1}$ is the prediction of x at $n+1$, E_t is the prediction error, and p_0 is the prediction of the last observation. The prediction error is given by the absolute value on the difference between the forecast and the t observation. The variance $\sigma^2(E_t)$ is calculated considering the prediction error at the prediction period t (i.e. the horizon of the estimation).

In addition, the thresholds include a parameter K, from which use case operators are able to adjust the sensitivity of the constraint. The default value of K is $Z_{\alpha/2}$ thus relating the thresholds with the normal distribution of the series. Note that this is not a wrong decision considering publications as [HKOS05], where it was shown that when the time series does not approach the normal distribution, the error is unrepresentative; in the case of the exponential smoothing algorithms, the margin rate of both intervals is in the order $100(1-\alpha)$, where α, $0 < \alpha < 1$, is the smoothing constant.

Based on these equations it is possible to deduce that the higher the value of K, the lower the level of restriction of the thresholds; therefore, the system will operate with greater tolerance to prediction errors (see Figure 4.12(a), where $K = 1.5$). In the opposite case, the system tends to infer a greater amount of facts related to the exceedance of some of the prediction thresholds (see Figure 4.12(b), where $K = 0.5$). Its configuration depends on the characteristics of the use case, and it is highly recommended its calibration by machine learning approaches.



(a) K=1.5 (b) K=0.5

Figure 4.12: Example of Prediction Intervals and K variations.

4.4 Final Remarks

This chapter reviewed the prediction and adaptive threshold methods applied on univariate time series for estimating forecasted observations. To this end, observations are modeled according to the mathematical approximation conducted by the reference forecasting method. Because of the importance of dealing with predictive capabilities when projecting the status of the network, as indicated by the Endsley model, some of these methods are conveniently applied in the forthcoming chapters of this research with the objective to perform data forecasting analysis.

Chapter 5

Pattern Recognition

This chapter introduces well-known pattern recognition algorithms that are intended to infer similarities between samples composed by grouped metrics, and datasets of reference samples. Because three main pattern recognition actions are distinguished in the literature, the chapter contents have been organized as follows: Section 5.1 outlines the pattern recognition landscape in networking. Section 5.2 describes the classification algorithms. Section 5.3 introduces common matching methods documented in the literature, whereas novelty detection approaches are presented in Section 5.4. Finally, Section 5.5 provides the concluding remarks of this chapter.

5.1 Pattern Recognition in Networking

The ability to analyze huge amounts of monitored data in emerging communication environments is crucial for accomplishing an efficient autonomic management. It strongly depends on the capability to disclose pattern and common behaviours on the sampled data, which is achieved by automating the classification of sampled data taken as input into a finite number of categories.

From the networking perspective, the need of searching patterns in data is fundamental in emerging contexts, hence gaining relevance in 5G. The patterns to look for are already available in the reference datasets used when supervised training mode is managed, which reduces the need to perform costly testing to disclose a particular pattern. This trait poses a major advantage for accomplishing autonomic management since the timely recognition of a given pattern can lead to enhance the decision-making process, and the consequent deployment of countermeasures in the monitored environment.

For instance, in the field of traffic classification, the research community has investigated classification approaches for inferring application-level usage patterns without the need to deploy Deep Packet Inspection (DPI) servers. It provides not only faster response time against pre-defined network threats, but also significantly less processing overhead in the network. Such approaches classify traffic patterns by identifying statistical patterns in traffic attributes such as the packet length or inter-packet arrival for categoryzing applications of interest for management purposes [NA08].

Complementary, matching and novelty detection methods have gained importance in

the analysis of modern network environments. Novelty detection methods determine wheter an observation belongs to the same distribution as the existing ones (inlier) or not (outlier), whereas matching aims to evaluate if a sample of observations is equal to any of the reference samples of a dataset. As is the case of classification, autonomic network management can take advantage of both novelty detection and matching in the inference of knowledge about the monitored environment. Thereby, the characteristics of those pattern recognition actions are described in the forthcoming sections.

5.2 Classification

This section focuses on the Decision Stump, RepTree, Random Forest, Bootstrap Aggregation, Adaptive Boosting, Bayesian Network and Naïve Bayes classifiers.

5.2.1 Decision Stump

The decision stump is a simple tree classification model that unlike other tree proposals in the bibliography, only performs one split, hence being commonly considered as a one-level decision tree [JST+07]. The decision stump approaches regression based on a single feature, which makes it not the most powerful classification tool but a simple an efficient solution, usually being adopted as component in machine learning ensemble approaches, such as bagging or boosting [FI92][SL91]. Figure 5.1 illustrates an example of classification based on decision stump. The algorithm concludes that the pEntropy feature is the most relevant, and built a one-level tree for deciding if the analyzed samples belong to the class "legitimate" or "botnet". In the example the hit rate was 84

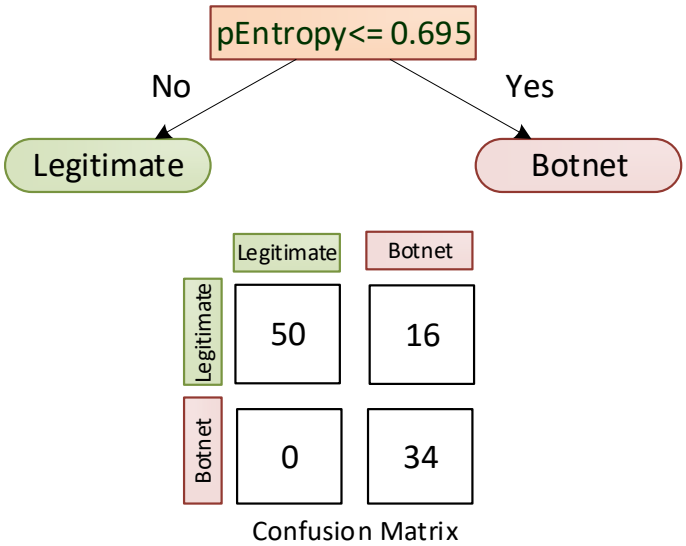


Figure 5.1: Example of classification by Decision Stump

5.2.2 Reducing Error Pruning Tree

RepTree is a simple tree-based classification method that lies in the Reducing Error Pruning for Tree construction (REPT) technique [SL91]. Note that REP is a simple pruning approach that creates tree-based classifiers by the following procedure: starting at the leaves, their nodes are replaced with the most relevant classes. If the accuracy of the resultant classifier is not worsened, the changes prevail. This approach reduces the size of decision trees by removing sections of the trees that provide little power to classify instances. RepTree builds decision/regression trees based on the principle of computing the information gain with entropy and minimizing the error arising from variance, and then pruning them by applying reduced-error pruning with backfitting [BF85]. The resultant classifier is a multi-level tree in which each level evaluates conditions of a particular attribute. Figure 5.2 illustrates an example of RepTree automatically plotted by the tool WEKA.

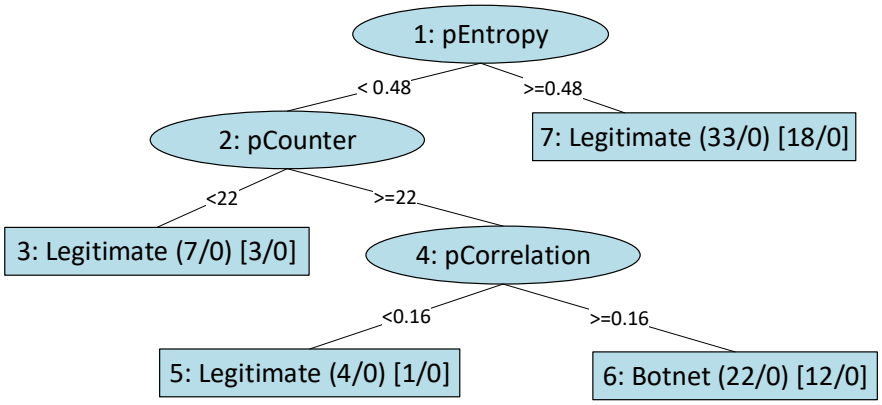


Figure 5.2: RepTree for Botnet detection

5.2.3 Random Forest

According with the random forest scheme proposed by Breiman [Bre01], a random forest is a classifier consisting of a collection of tree-structured predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. In particular, the original approach implemented the variation of Classification And Regression Trees (CART) [RJPD14a] that choose which variable to split on using a greedy algorithm that minimizes error. The final classification of a *PRSample* container comes from the correlation of the classifications emitted by all the generate trees; in this way, the likelihood of a sample belonging to a class is the probability of being emitted as a classification by some of the trees (see Figure 5.3).

Note that this process requires the specification of several adjustment parameters such as the maximum amount of iterations to be performed, number of trees or their maximum depth. However, as highlighted by Breiman, the number m of randomly selected attributes is the only adjustable parameter to which random forests are somewhat sensitive. This value determines the correlation between each pair of trees and the strength

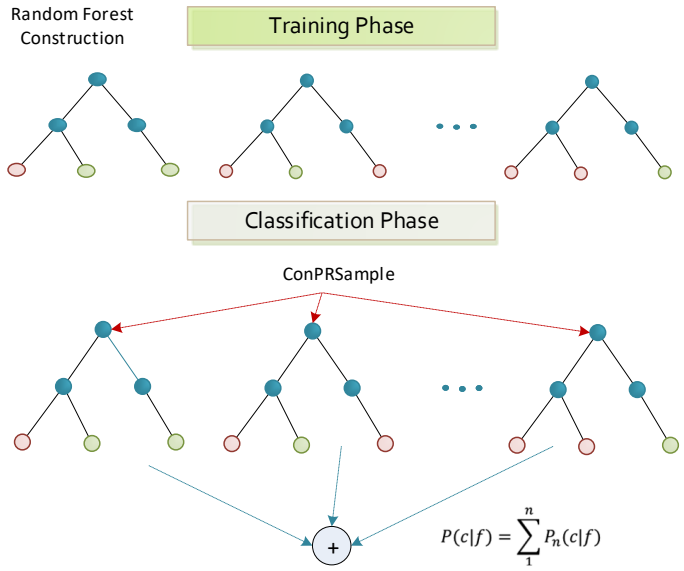


Figure 5.3: Random Forest

of each individual tree. By increasing the aforementioned parameter, both correlation and strength increase. When the correlation grows, the forest error rate increases; in the opposite, when strength grows the forest error rate decreases, so the level of both features must be balanced.

5.2.4 Bootstrap Aggregation

Bootstrap Aggregation, also refereed as *bagging*, is an ensemble algorithm with classification and regression capabilities. In the implementation of bagging, based in the proposal of Breiman [Bre96], bootstrap is adopted as statistical estimation technique where certain statistical indicators (i.e. mean, variance, etc.) are from multiple random samples within the reference dataset with replacement. Breiman extended this approach to machine learning models. In this way, different machine learning models are built from multiple random samples of the training dataset (hence the term *bagging*) (see Figure 5.4).

When classifying samples, the results of the different classifiers are aggregated in order to provide a unified solution. Therefore, this is a scheme very similar to Random Forest, but with a major distinction: in Bootstrap Aggregation all features are considered for splitting each node. On the opposite, in Random Forests only a subset of the attributes is selected at random out of the total. Then the best split feature of the subset is applied to split each node in a tree.

5.2.5 Adaptive Boosting

Adaptive Boosting [ROM01], also refereed as AdaBoost, is a machine learning algorithm which inherits the idea of combining different classifiers observed in Random Forest or Bagging, but with the nuance that the results obtained by a classifier are taken into account when building the next one (boosting). Adaptive Boosting was originally designed to

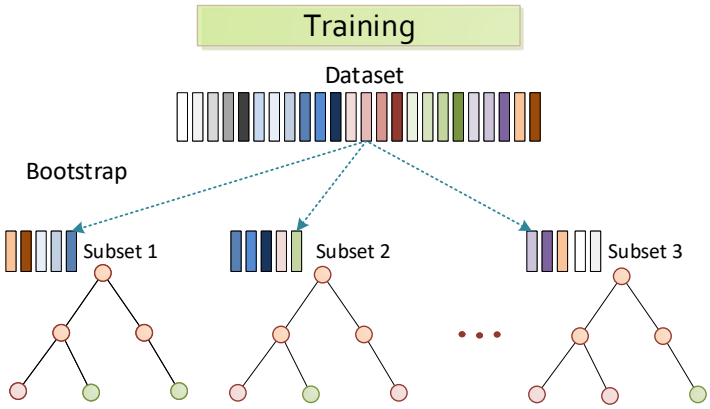


Figure 5.4: Training in Bootstrap Aggregation

consider simple decision tree classifiers, each with a single decision point (usually decision stumps). At training stage, each sample in the training dataset is weighted and the weights are updated based on the overall accuracy of the model and whether an instance was classified correctly or not (see Figure 5.5). In particular, a weight w_{th} is assigned to each sample within a reference dataset equal to the current error $E(F_{T-1}(x_i))$ on that sample, where F_{T-1} is the previous classifier and x_i the input in position i. This process is repeated until certain minimum error is achieved or when it is not possible to improve the obtained results with new training steps. The classification consider the decision made by every classifier, but weighted according to their weights, so the final output of the system is obtained as a weighted linear combination of all base classifiers.

5.2.6 Bayesian Network

Bayesian Networks are the graphical representation of sets of random variables and their relationships by means of acyclic directed graphs, in which the vertices are the observed data and the edges their conditional probabilities. As indicated by Buczak and Guven [BG16], these structures can be constructed by experts or algorithms based on inference, which constitute the modelling stage of the system. An example of a Bayesian Network is illustrated in Figure 5.6, where vertices are sample features (ex. communication delays, bandwidth, number of sessions, etc.) and edges are the probability that one observation involves a dependency with another according to the Bayes theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{5.1}$$

where A and B are the monitored features. There are different approaches for building Bayesian Networks with classification capabilities [WkB], distinguishing two major model building steps: structure learning and probability estimation. The first stage aims on defining the Beyesian Network structures, being supported the different sets of methods: local and global score based algorithms, conditional independence tests, or even loading a predefined structure.

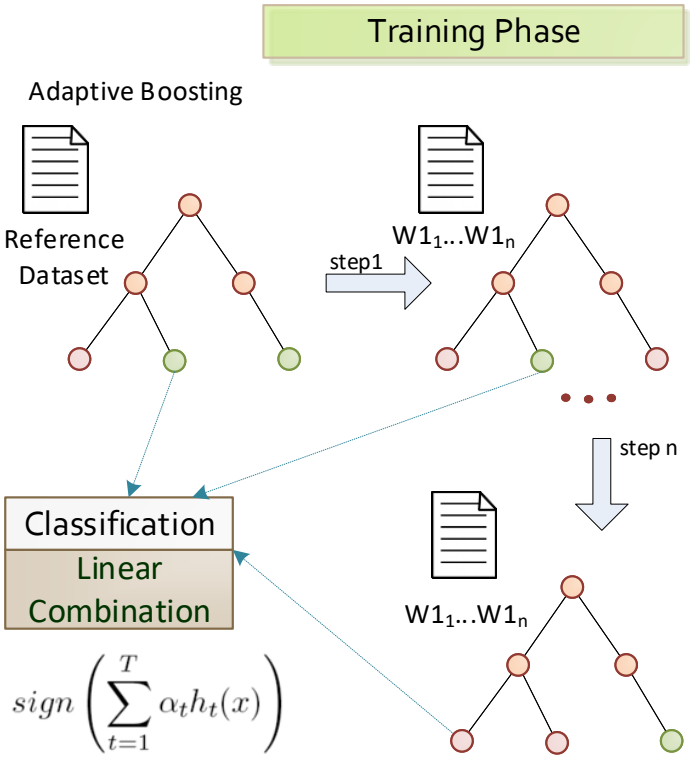


Figure 5.5: Adaptive Boosting

5.2.7 Naïve Bayes

Naïve Bayes [JL95], is not a simple method, but a family of algorithms based on the Bayes theorem sharing a common principle: every feature being classified is independent of the value of any other feature (hence the adjective naïve). Because of this, Naïve Bayes classifiers are considered Bayesian Networks with a simple structure that has the class node as the parent node of all the attribute nodes (see Figure 5.7). This characteristic makes them much simpler and faster than the conventional Bayesian Networks. On the other hand, if the conditional independence assumption actually holds, the classifier will converge quicker than Bayesian Networks, hence requiring less training data.

5.3 Matching

Matching has the goal to test if a sample of observations is equal to any of the samples contained in a reference dataset. Note that unlike classification actions, matching does not take into account the degree of similarity between the monitored data and the reference samples or the class they belong; only if the sample appears as it is in any of them. Hence the class attribute of the ARFF dataset files is ignored.

5.3.1 Dictionaries and Bloom Filters

Bloom filters [RK15] are probabilistic data structures used to determine whether a data belongs to an impractically large dataset or not. Their main features are enormous

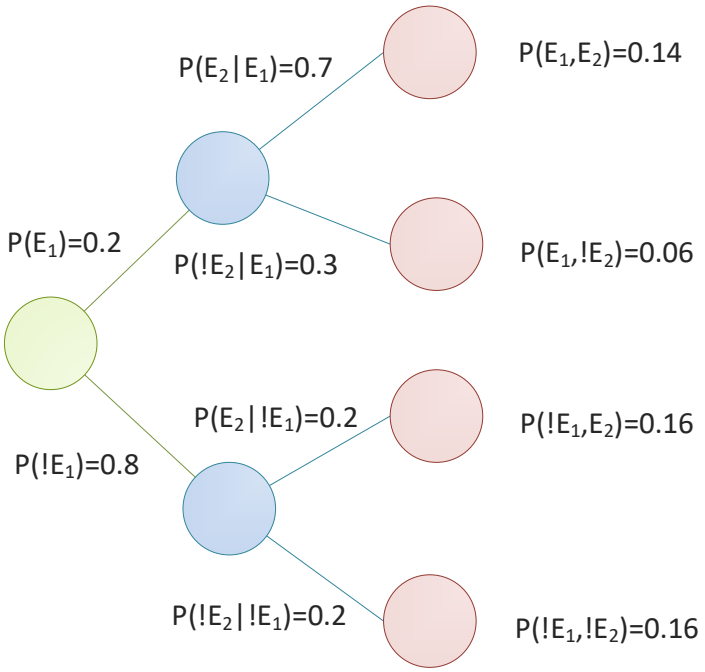


Figure 5.6: Bayesian Network

efficiency ($\theta(k)cost$), low memory consumption and non-generation of false negatives, which are achieved at the cost of storing only the information strictly necessary to determine if an observation was previously observed. Note that when datasets are small, they are easily managed by conventional data structures (lists, maps, etc.). But when they are very large, Bloom filters are one of the few structures capable of handling all the information they contain for their specific purpose. Because of this Bloom Filters are frequently applied in networking with different purposes, among them pattern matching and Deep Packet Inspection (DPI) [RL06] [GVSOMV17].

The most common representation of a Bloom filter is an array of m bits with k different hash functions, each of which maps or hashes some set element to one of the m array positions, hence generating a uniform random distribution (see Figure 5.8). Given the probabilistic nature of this data structure, the use of a number of inappropriate hash functions and operating over a too large sample space may lead the Bloom filter to generate false positives. This occurs when collisions occur when consulting filter records, i.e. the dispersion function redirects the search to two or more different positions which have different values. The collision is due to the fact that the just identified element has not been observed before, but when previously registering another observation, some of the positions that represent the new one was modified.

The best Bloom filter setting is approached based on the following probability of issuing false positives [GA13]:

$$TFP = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k \approx \left(1 - e^{\frac{-kn}{m}}\right)^k \tag{5.2}$$

where n is the number of elements to classify, m is the number of bits that identify the

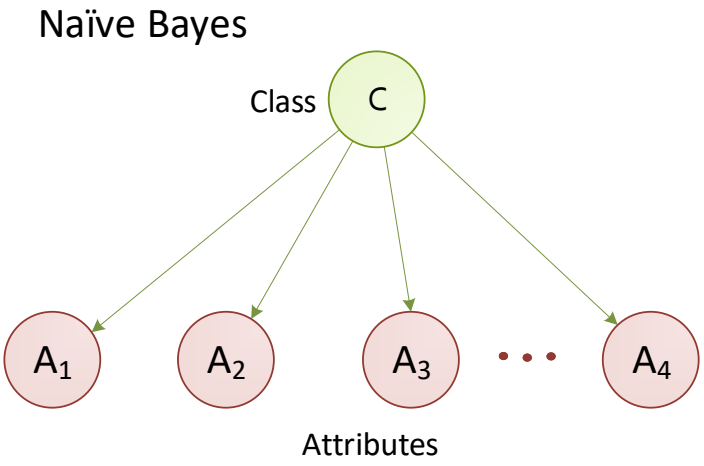


Figure 5.7: Naïve Bayes classifier

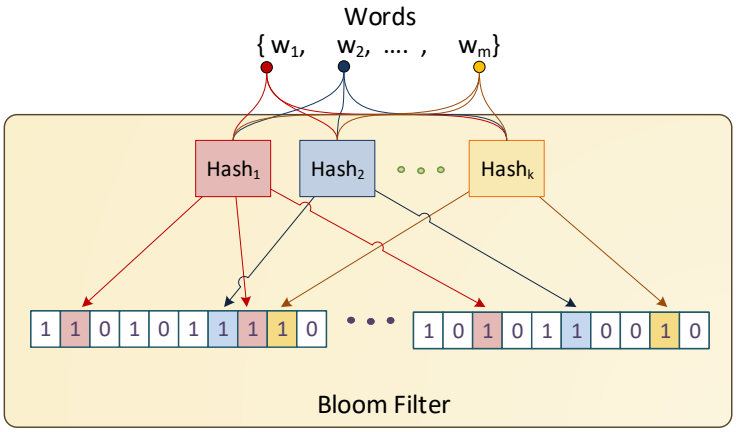


Figure 5.8: Bloom Filter

Bloom filter slots, and k is the number of dispersion functions. Bearing this expression in mind, it is possible to calculate the optimal amount of dispersion functions for a particular sensor according to its specification. This is summarized as follows:

$$K = \frac{m}{n} \times ln2 \tag{5.3}$$

Alternatively, a simple way to enhance the Bloom filter hashing is considering universal hash functions. But their implementation often involves the increase of m, in this way penalizing the size of the representation in memory of the structure.

5.4 Novelty Detection

Novelty detection is usually defined as the task of recognizing that test data differ in some respect from the data that are available during training, which also can be refereed as one-class classification [PCCT14]. These methods are usually applied to solve problems where the analytic system was provided by a long and complete collection of reference

samples (commonly "normal" observations), and it is required to decide if the observed data can be tagged as belonging to the population on the reference dataset, or it has "discordant" nature. For example, let the dataset illustrated in Figure 5.9 which contains instances characterized by basic IPFIX features (*source*, *destination* and *nPackets*) and some aggregated metrics calculated from the packets they represent (*pEntropy* and *NLevenshtein*).

```
@relation habitualTraffic
@attribute 'source' string
@attribute 'destination' string
@attribute 'nPackets' numeric
@attribute 'pEntropy' numeric
@attribute 'NLevenshtein' numeric
@attribute 'class' {habitual}
@data
"192.16.25.13","162.65.30.04",50,0.45,0.48,habitual
"192.16.25.18","162.65.30.06",115,0.65,0.26,habitual
"192.16.25.19","162.65.30.20",65,0.71,0.84,habitual
"192.16.25.20","162.65.30.06",47,0.53,0.89,habitual
"192.16.25.13","162.65.30.06",260,0.63,0.83,habitual
"192.16.25.19","162.65.30.04",179,0.65,0.54,habitual
"192.16.25.13","162.65.30.20",135,0.48,0.61,habitual
"192.16.25.18","162.65.30.04",78,0.73,0.97,habitual
"192.16.25.20","162.65.30.31",56,0.71,0.34,habitual
"192.16.25.13","162.65.30.06",93,0.59,0.61,habitual
"192.16.25.13","162.65.30.32",165,0.51,0.31,habitual
"192.16.25.18","162.65.30.04",219,0.53,0.47,habitual
"192.16.25.19","162.65.30.04",194,0.64,0.44,habitual
"192.16.25.20","162.65.30.06",188,0.52,0.42,habitual
"192.16.25.13","162.65.30.20",73,0.71,0.51,habitual
"192.16.25.19","162.65.30.37",56,0.35,0.53,habitual
```

Figure 5.9: Example of one-class dataset

By novelty detection it is possible deduce if the following samples belong to the same class than the reference data:

$$Flow_1 : \{192.16.25.13, 162.65.30.04, 70, 0.5, 0.83\}$$
$$Flow_2 : \{192.16.13.13, 162.65.04.04, 48, 0.42, 0.46\}$$
$$Flow_3 : \{192.16.19.13, 162.65.06.04, 72, 0.72, 0.71\}$$

If they are tagged as "normal", it is possible to state that monitored data is similar to the normal traffic on the network. In the opposite, the flows are anomalous and probably require in-depth study. The following describes two main methods for novelty detection: Support Vector Machines and classification generating synthetic data.

5.4.1 Support Vector Machines

The Support Vector Machines (SVM) are a collection of supervised automatic learning algorithms based on the transformation of the input space into another one of superior and infinite dimension, where the problem to be dealt with is solved from the calculation of the optimal hyperplane called support vector [BL02]. Hence, given a reference dataset

(plane) represented as a *p-dimensional* sorted vector, Support Vector Machines calculate the hyperplane that optimally separates the samples into classes, thus defining the sets of membership (See Figure 5.10).

The ideal hyperplane for class delimitation is a *1-dimensional* vector which maximizes the distance between the members of each class, usually refereed as maximum-margin hyperplane. However, it is not always possible to find, mainly due to the compensation of errors registered at training steps or computational limitations. The latter usually leads to the implementation of predefined kernel functions (ex. homogeneous polynomial $k(\vec{x_i}, \vec{x_j}) = (\vec{x_i} \times \vec{x_j})^d$, perceptron $k(\vec{x_i}, \vec{x_j}) = \| \vec{x_i} - \vec{x_j} \|$, etc.), which project the information to a space of greater dimensionality on which it is possible to operate more efficiently.

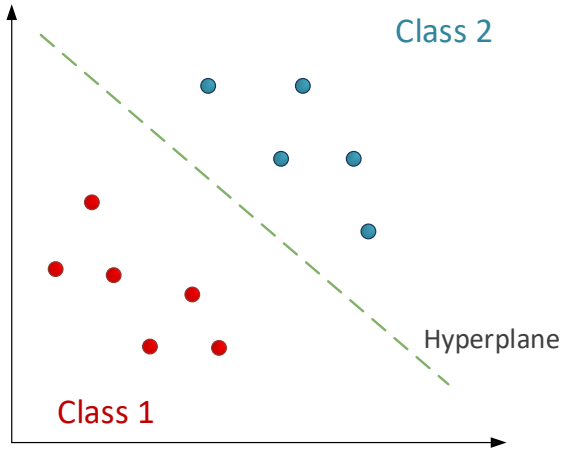


Figure 5.10: Two class Support Vector Machine

5.4.2 Generation of synthetic data

The novelty detection method for one-class classification, proposed by Hempstalk [HFW08], combines the application of a density estimator, used to form a reference distribution, with the induction of a standard model for class probability estimation. As is common in the bibliography, the reference distribution is used to generate artificial data that is employed to form a second synthetic class where this artificial class is the basis for a standard two-class learning problem. According with this publication, the combination of both density and class probability estimation are merged into the following curve:

$$P(X|T) = \frac{(1 - P(T))P(T|X)}{P(T)(1 - P(T|X))} P(X|A) \tag{5.4}$$

where T is the class of the sample in the reference dataset, A is the class of the artificial data to be built, and $P(X|A)$, $P(X|T)$ are their density distributions. For instance, the tool WEKA implements 10 iterations of bagged unpruned RepTrees decision trees with Laplace smoothing as the probability estimator $P(X|T)$, and a product of mixture of Gaussian distributions with one mixture per attribute [SG99] as density estimator where each mixture is fitted to the target data for its corresponding attribute using the Expectation–Maximization (EM) algorithm [Moo96]. The number of the artificial

sample matches the size of the reference dataset. Hence the data used to build the bagged unpruned decision trees was exactly balanced. Once the artificial samples are created, a two-class classifier is built assuming both sample collections.

5.5 Final Remarks

This chapter reviewed the pattern recognition methods to be applied on samples of observations as part of the knowledge acquisition process conducted to grant self-organizing capabilities on the monitored network. The inclusion of patter recognition capabilities is aligned with the comprehension and projection stages of the Endsley model, as it is the case of prediction methods. Consequently, the conducted research takes benefit from both.

Chapter 6

Knowledge Acquisition Framework for 5G Network Analytics

This chapter proposes a framework aimed on granting analysis capabilities to 5G networks by the proper instantiation of its components to achieve autonomic management, thus decoupling data analysis logic from specific data extraction procedures carried on at the lowest architectural levels of the network. It allows the insertion of several prediction algorithms, pattern recognition capabilities and production rules to generate meaningful knowledge that will enhance decision-making processes from the performance and efficiency perspective. Thereby, the framework provides the advantage to express metrics gathered by sensors (initial facts) according to a knowledge representation language in order to deduce conclusions about possible network scenarios driven by the Endsley approach [BLVCMV+17]. Perception, comprehension and projection steps are performed to understand the system state. The discovery of initial facts, which corresponds to previously monitored data, accomplishes the perception step. Reasoning involves both perception and comprehension, whereas prediction approaches the projection step. The deduced final facts (conclusions) are described in the form of symptoms related with each use case. Bearing this in mind, it is possible to assert that this framework provides a symptom-oriented situational awareness bounded by the configuration defined for each use case.

The major contributions of this proposal on the advances on knowledge-based management approaches upon 5G infrastructures are summarized as follows:

- *A novel reasoning 5G-oriented architecture.* A novel framework composed by functional elements arranged on an orchestrated workflow is proposed to enable reasoning capabilities in a 5G network. As a result, the framework generates conclusions about the 5G network status. The introduced architecture distinguishes two types of knowledge: procedural and factual. Procedural knowledge corresponds to the use case configuration loaded to the system. Initial factual knowledge is acquired by discovery methods applied on data previously collected by several sensors distributed along the 5G network, whereas factual knowledge is generated by the prediction, pattern recognition and knowledge inference modules introduced in this

proposal. Thereby, the framework approaches the perception, comprehension and projection steps of the Endsley model.

- *Instantiation of the framework.* An instance of the proposed framework has been created to enhance the understanding of the proposal. To this end, well-known multiplatform open source technologies and a battery of prediction and machine learning algorithms have been integrated in accordance with the framework design. In addition, publicly available datasets were applied to allow its experimental replicability. The generation of knowledge was successfully demonstrated in a datacenter-oriented use case, but the current instance of the framework can be applied on several use cases just by modifying its configuration.

- *Comprehensive experimentation on a real use case.* To assess the accuracy of the instantiation, a set of experiments have been conducted. They were oriented either for the evaluation of the pattern recognition and prediction modules; and for the evaluation of a real use case. Prediction and pattern recognition features exposed good accuracy levels when applied over the reference datasets. Likewise, a particular use case configuration to generate conclusions about traffic behaviour has been tested. The experiments were conducted on real network traffic samples where the inference of suspicious network traffic volumes in a datacenter exposed good precision rates contrasted with the real reference scenario.

To facilitate the understanding of the proposal, this chapter has been divided into six sections. Section 6.1 describes the design principles and constraints assumed for this proposal. Section 6.2 introduces the framework architecture and a detailed explanation of its components. In Section 6.3 an example of the framework instantiation is presented. Section 6.4 presents the experimentation conducted for validating the proposal. Section 6.5 discusses the results obtained by the experimentation. Finally, Section 6.6 remarks the conclusions of this chapter.

6.1 Design Principles and Constraints

The following design requirements and assumptions have been kept in mind at both design and implementation stages of the reasoning and knowledge acquisition framework.

- *Scalability.* The proposed framework must accommodate the 5G design principles, and in particular, those associated with scalability, such as "Extensibility by design", "Expandability by design" or "Multi-level scalability by design" [NCC+16], through the combination of scalable modular design, open interfaces and APIs to enable third parties to create their own automatic network management services.

- *Support of use case onboarding.* The knowledge acquisition framework adopts a use case driven research methodology. Because of this it is required that from design, it must support the onboarding of new different use case specifications. Given the heavy reliance of the tasks performed with the characteristics of use

cases, the basic definition of the observations to be studied (knowledge-base objects, rules, prediction metrics, etc.) must be provided as factual knowledge by use case operators, thus being the framework scalable to alternative contexts. In addition, use case operators must provide procedural knowledge, thus configuring the analytic tasks to be performed per use case.

- *Reference datasets.* Laskov et al. [LDSR05] realized two essential observations necessary to understand the different strategies for acquiring reference knowledge and to decide the most appropriate for each use case: firstly, it must be taken into account that labeled samples are very difficult to obtain, a situation that can be aggravated if the sensor operates in real time, and/or on monitoring environments where is not possible to extract all the data; on the other hand, there is no way of collecting labeled samples which cover every possible incident, so the system is potentially vulnerable to unknown situations. To these difficulties it is added the problem that there are no collections of traffic captures in 5G networks, and that the existing datasets of current traffic traces often have drawbacks such as lack of completeness or labeling errors. Because of this, the proposed framework does not go deep into the issue of the innate knowledge acquisition. The current approach assumes that the reference datasets are provided by skilled operators or by accurate machine learning algorithms, which therefore are valid for the specified use cases.

- *Granularity.* 5G environments are complex monitoring scenarios where large amounts of sensors collect information about the state of the network in real time. In SELFNET all this information is processed in the Aggregation sub-layer, which provides the necessary metrics to infer knowledge from them. However, this information is not raw processed. As described in [LVV17], it is compiled into Aggregated Data Bundles (ADBs), which summarize all the system information observed over a time period related with the previously declared uses cases. The length of the observation period defines the data granularity, which may be determinant for the proper functioning of certain uses cases. However, the decision of the best granularity is out of the scope of this proposal.

- *Stationary monitoring environment.* By definition, the features monitored on a stationary scenario are similar to those considered when building data mining models. The assumption of operating on a stationary monitoring environment entails ignoring in terms of learning process any variation in the characteristics of the information to be studied, such as dimensionality or distribution. The main disadvantage of this approach is the loss of precision when such changes occur, in large part because the initially performed calibrations are not adapted to the current status of the network. On the other hand, their proper accommodation tends to retain the acquired calibration at the expense of addressing many other issues, emphasizing among them to discover relevant variations in the data nature, calibration upgrades based on the new features, or improvement of the original datasets [DRAP15]. Being aware that the last approach poses important challenges, and in order to facilitate the understanding of the proposed research, all those aspects

related to the management of the non-stationary characteristics of the information are overlooked.

- *High dimensional data.* When the dimensions of the data to be studied are more extensive than usual, it is possible that some reasoning and knowledge acquisitions implementations lose effectiveness, either in terms of efficiency or accuracy. Because of this, the bibliography provides a wide variety of publications focused on the optimization of this kind of processes, as is discussed in [AY01]. Note that the battery of algorithms included in the current instantiation of the framework does not adapt any of these contributions, which does not mean that it is incompatible with them. However, throughout the document the risks of operating with high dimensional data are not taken into account, in this way postponing this problem to future instantiations.

- *Software Security.* Operating in a trusted environment is a challenging task that should be addressed by the incorporation of the best software development practices for both design and implementation. Even though the instantiation of this framework involves the integration of different software components, security considerations in terms of software development and network communications are overlooked since experimental validation is prioritized.

6.2 Architecture

The proposed framework is composed by the functional elements illustrated in Figure 6.1. They are settled down to interact as providers and consumers of facts not only discovered from monitored data, but also deduced by reasoning procedures. The architectural elements of the framework are pipelined sequentially as: Onboarding of use cases, Discovery, Pattern Recognition, Prediction, Adaptive Thresholding, Knowledge Inference and Notification. These elements are coordinated by the orchestration strategy defined in [LVV17]. Hence, and assuming the design principles previously stated, the proposed framework brings analytic capabilities focused on acquiring knowledge from the network metrics (initial facts), and deduces conclusions (final facts) such as likelihood of the network being attacked, anomalous congestion levels, among others.

The Discovery component obtains information, represented as facts, gathered by network sensors in the lower levels of a 5G architecture by monitoring, data aggregation and correlation procedures. It exempts the framework to the need of dealing with network technology-dependent protocols or interfaces, and allows assuming that the constraints inherent in the monitoring environment (for example, ciphering, privacy protection politics, etc.) have no impact on the effectiveness of the proposal, since they have been previously managed at lower data processing stages. New knowledge is acquired by the Inference Engine based on the collected facts stored in the Working Memory, and permits the inference of conclusions about the network status by applying production rules configured in the Knowledge Base. Conclusions are expressed as symptoms, reflecting situations that might affect or compromise network operability or a degradation of the

agreed service levels. The framework also facilitates a situational awareness projection of the network through the Prediction and Adaptive Thresholding components, by calculating predictive metrics and forecasting intervals that allow pro-action responses over the predicted scenarios. Likewise, the Pattern Recognition component implements some of the recent paradigms of Artificial Intelligence, among them machine learning, data mining, classification or novelty detection methods.

To deal with scalability, each analytic component is designed to be run independently, exchanging only input and output data between them through buffering data structures or message broker tools (such as Apache Kafka [Akf] or RabbitMQ [RbM]), thus decoupling processing tasks. The underlying technology to instantiate each component (i.e. Weka, Drools, among others) must also be able to scalable by design (vertically or horizontally) to accommodate several deployment strategies when computing resources demands are variable. In this way, the proposed framework is aligned with the principles of 5G by developing a scalability solution adapted to the latest trends in the control and data planes of 5G mobile architectures [EtM], allowing its deployment on the management side. The role of each framework component is explained in detail in the following subsections.

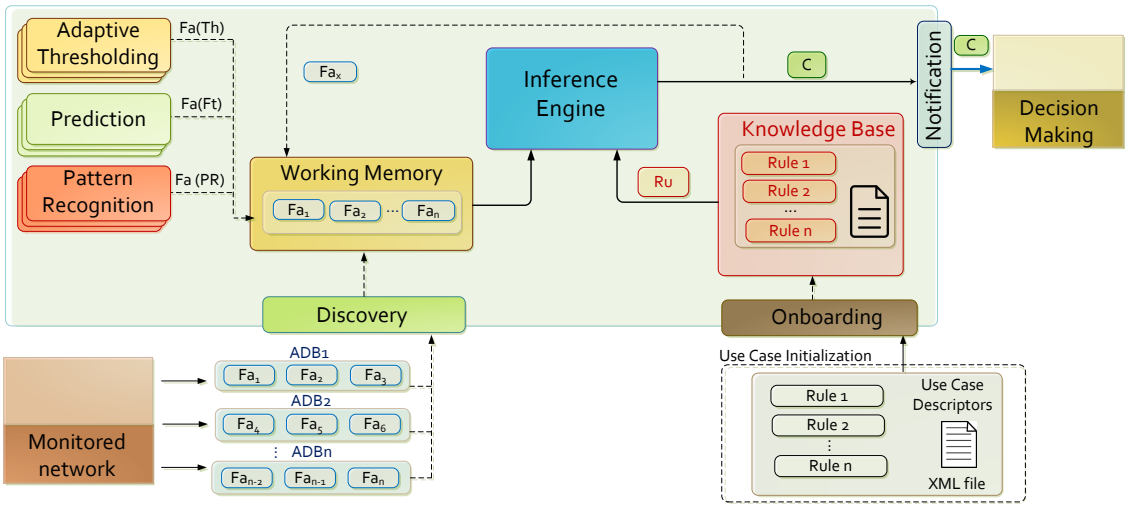


Figure 6.1: Knowledge acquisition framework for 5G environments.

6.2.1 Initial knowledge and Notifications

The Knowledge Base is filled from data acquired from the use case definitions at use case Onboarding. Because of this, use case operators may declare procedural knowledge such as inference rules Ru, prediction actions Ft, etc. and specify factual knowledge such as Objects O, Facts Fa, Thresholds Th, etc. in compliance with the use case descriptors defined in [Bar17]. They also provide the reference datasets required for machine learning actions. Factual knowledge is gathered by the Discovery component, which periodically receives ADBs which summarize the acquired observations. From the loaded metrics and events, the knowledge acquisition framework builds facts (Fa). If they are required for prediction, pattern recognition or adaptive thresholding, these observations are inserted in

the temporally stored time series. Note that independent facts are removed at the end of the ADB processing, as well as the new knowledge acquired from them. It is remarkable that the abovementioned procedural and factual knowledge represent the inputs of the proposed framework. However, it is worth mentioning that new factual knowledge is also internally generated by the Prediction, Pattern Recognition and Adaptive Thresholding components for the inference of new knowledge.

The set of actions on Notification informs the final facts (conclusions) acquired by the framework. This step packs the conclusions by inserting or modifying the related meta-knowledge to accommodate contextual information, such as facts location, timestamps, output format representation (i.e. JSON), among others. Once an ADB is fully analyzed, these actions also erase and restart the auxiliary functionalities on the analytics and several data structures. Only the information and buffers required for building time series with data to be extracted from future ADBs is temporally persistent.

6.2.2 Prediction Module

This component drives the inference of knowledge related with prediction facts built from the monitored data. Its main purpose is to insert facts about forecasted metrics in the Working Memory, and the observation of variations of interest such as discordant values or relevant decreases or increases on the analyzed data. Although the framework does not support persistent storage, several data must be temporally preserved to allow registering time series and information needed for enhancing the decision and calibration of the prediction algorithms. Then the forecasting strategies must be cautiously selected and adjusted with the purpose of providing the more accurate results. Once the predictions are calculated, the system includes the discovered knowledge (facts) into the Working Memory. Note that this data processing stage depends on synchronous ADB loading where time series are fed with observations fetched from the ADBs. It requires two types of knowledge: procedural and factual. Procedural knowledge is provided via use case onboarding in their data descriptors. Likewise, factual knowledge is acquired by the Discovery component, when new ADBs are loaded. Once the time series with the required length are built, several prediction methods are evaluated to decide the most accurate algorithm fitted to the given time series. The selection of the best forecasting methods entails several steps. Once data is acquired, a time series of size N, and the forecasting horizon T are taken as input parameters for a preprocessing task. The last T elements are subtracted from the original time series and the remaining $N - T$ elements are used for forecasting. In the meantime, the subtracted T elements are reserved to be used for evaluation. Parameter calibration takes place for all the forecasting algorithms included in the framework, each one has a variable number of parameters. Every individual parameter can be tested with different values, thus allowing a forecasting algorithm to be run with different calibration coefficients.

6.2.3 Adaptive Thresholding Module

Throughout the tasks involved in the reasoning process, it is necessary to define under what circumstances an observation about the monitoring environment must conditionate the inference of new knowledge. This is a complicated challenge, which must take into account both the situational awareness of the monitoring scenario and the specific use cases on which the self-organizing deployment for incident management has been implemented. Therefore, the instantiated adaptive thresholding strategies must pose dynamic solutions, subject to the changes in the different features of the scenario on which they operate, and must be configurable according to the level of constraint that operators decide (note that the restrictiveness may also be stablished by machine learning approaches). Because of this, the calculated thresholds act on any source of knowledge of the Knowledge Inference engine (e.g., Discovery, Prediction, Pattern Recognition, etc.), or may be part of the production rules. This makes the results they provide considered as factual knowledge by the knowledge-based of the framework, being Adaptive Thresholding and additional knowledge acquisition step dependent of the rest of the components of the proposal.

6.2.4 Pattern Recognition Module

The Pattern Recognition component of the proposed framework operate at two different stages: training and discovery. At training, the knowledge representation to be taken into account, as well as the description of the pattern recognition actions are included into the procedural knowledge according to the use case specifications. This step involves generating/loading reference datasets and construction of the best models in function of the most relevant data features on the sets of metrics to be analyzed. The training step may take place in two moments of the analytic process. Firstly, models from reference data can be built before operating on real monitored samples. On the other hand the training step may operate at runtime, so the reference samples are gathered from the first observations on the monitoring environment. At the discovery stage, the knowledge acquisition framework launches the pattern recognition actions defined by the use case operators. Samples are constructed from the aggregated metrics observed, and they are analyzed based on the models built at training stage. The framework at least allows two pattern recognition actions: classification and novelty detection. When classifying, a reference dataset is loaded before the monitoring of the protected environment. A battery of classification algorithms is executed in concurrency, which are properly calibrated and combined as an ensemble of models [Zim14][Rok10]. Then the most accurate classifier is identified by cross-validation on the reference sample collection [K$^+$95], and it is applied at the discovery step. On the other hand, the novelty detection actions are usually defined as the tasks of recognizing that test data differ in some respect from the data that are available during training, which also can be generalized as one-class classification [PCCT14]. These methods are usually applied to solve problems where the analytic system was provided by a long and complete collection of reference samples (commonly “normal” observations), and it is required to decide if the observed data can be tagged as belonging to the population on the reference dataset, or if it has “discordant” nature. The proposed

framework implements novelty detection similarly to classification, in this way also based on an ensemble of sensors. However, in this case, the training stage considers data observed at runtime from the monitoring environment. Particularly, the first monitored metrics define the reference dataset, and hence are tagged as normal observations. The length of this collection is previously defined by the use case operators. It can be manually delimited or decided by the results of the cross-validation scheme; in the second case, if the accuracy is greater than certain threshold, the model is considered acceptable and there is no need to process additional samples.

6.2.5 Knowledge Inference Module

The knowledge inference component allows deducing information from previous observations (facts) based on procedural knowledge represented as rule sets. In order to align the decision strategy of which rules should be activated and when, with the previously assumed design principles and requirements, the implemented approach is driven by production rules. This facilitates the deployment of a modus ponens (forward chaining) decision scheme where attributes enable the deduction of goals, which are final facts encapsulated as symptoms before their report to the decision-making layer. It should be kept in mind that throughout the bibliography, Rete algorithms are the most popular and proved proposals to address the efficient implementation and execution of forward chaining on complex monitoring environments. Created by Forgy [For82] , these methods separate the rule management into two steps: rule compilation and runtime execution. The first stage describes how the rules in the Working Memory are processed to state an efficient discriminant network, where upper nodes tend to present many matches, in contrast with the lower elements (the bottom are termed terminal nodes). The main reason on building this structure is to optimize the number of triggered rules, while at runtime, the previously built network allows inferring the new knowledge. Thereby, Rete algorithms are appropriated to address the knowledge inference purposes of the proposed framework.

6.3 Instantiation

As an illustrative example of instantiation of the proposed framework, this section describes how it has been deployed with the aim on contributing with the management of a real network. Its contribution is focused on the recognition of discordant behaviors based on analyzing the variations of the traffic volume, which borne in mind the prediction of their evolution, the construction of adaptive thresholding to decide when they may be considered unexpected, and novelty detection based on several distances and similarity measures. It is important to emphasize that the instantiated solution could be replaced by an alternative implementation and still achieving similar results. Nevertheless, this instantiation aims to describe a simple, didactic and scalable solution, that provides a greater understanding of the proposed framework and draft several basic guidelines for its adaptation to other problems. With this purpose, the following introduces the

implementation of each reasoning and knowledge acquisition component considered at the experimentation stage.

6.3.1 Initial Knowledge and Notifications Implementation

The initial knowledge of the instantiated framework is directly provided by the use case operators, hence postponing auto-calibration and other machine learning approaches for future work. It includes specific information about what activities must be monitored, what knowledge acquisition actions should be accomplished and what kind of reasoning must be carried out; the latter is guided by production rules and the inference of conclusions about the state of the network. Therefore, it can be said that the initial knowledge is the configuration of the framework and the strategy of acquiring the initial facts to be analyzed. These facts arrive to the system compiled as ADBs, which gather the information monitored in certain time periods. As stated before, the different sensors deployed on the 5G infrastructure are considered the most important information providers. For this framework instantiation the initial metric to be studied is the traffic volume per observation, which is assumed to be already reported by the sensors. With this collected data, the framework creates the required time series, enabling the possibility to apply prediction methods, i.e., to estimate the traffic volume at the coming observations according to a given forecasting horizon. The system is also configured to build adaptive thresholds based on the forecasts. They allow to identify if the observed traffic volume differs significantly from the predictions. Herein, those traffic observations are labeled as *unexpected*. On the other hand, the pattern recognition component is configured for novelty detection based in analyzing difference distances and similarity measures between each monitored observation and that of an immediately preceding monitoring period. Note that his action requires building one-class classification models, which demand a reference dataset. It is obtained from the first observations made, so directly loading an external dataset is not required. The discordant observations are in this stage labeled as *fluctuations*. On the other hand, the Knowledge Inference component is configured by production rules to conclude that an observation marked as *unexpected* and *fluctuation* is a suspicious event, hence being notified as a symptom. The findings are reported through a message broker software to be consumed by external sources, i.e., to perform more complex decision-making procedures. Table 6.1 summarizes the initial knowledge and notifications of the proposed framework instantiation considered at the experimentation.

6.3.2 Prediction Implementation

The current instantiation of the proposed framework does not support objects with multiples values. Because of this, the adapted battery of forecasting algorithms only considers univariate time series. This does not mean that this capability cannot be included in future instantiations, but it has been considered that working with a simpler instantiation facilitates the comprehension of the prototype, as well as the specification of new use cases. Two families of well-known forecasting methods are implemented: moving averages and exponential smoothing, as detailed in Table 6.2. They process the time series

Table 6.1: Summary of the instantiated initial knowledge and notifications.

Initial Factual Knowledge	
Element	**Description**
Object	Traffic volume monitored per sensor (V_t).
Acquisition	Via ADB.
Procedural Knowledge	
Component	**Behavior**
Prediction	Forecast traffic volume (V_t) given a certain prediction horizon.
Adaptive thresholding	Construction of decision thresholds from forecasted metrics.
Pattern Recognition	Novelty detection based on several distances and similarity metrics related with V_t and V_{t-1}.
Knowledge Inference	It is deduced that if Vt observations exceed adaptive thresholds, they are *unexpected*. If Vt observations are considered novelties, it is deduced that they are *fluctuations*. If V_t is *unexpected* and *fluctuation* then it is *suspicious*.
Notification	Reports *suspicious* events are reported via message broker software.

of monitored metrics in concurrency, and the decision of the best algorithm grounds on considering the minimum Symmetric Mean Absolute Percentage Error (sMAPE) [Mak00] as the forecasting error measure. sMape considers the set of real subtracted $N - T$ values X and the T forecasted values as inputs. It is described by the following expression:

$$sMAPE = 200\% \sum_{t=1}^{n} \frac{|F_t - X_t|}{|F_t| + |X_t|} \tag{6.1}$$

where X represents the real time series values and F are the forecasted values estimated for the given observations.

6.3.3 Adaptive Thresholding Implementation

To evaluate the prediction errors, two adaptive thresholds are constructed: an upper threshold Ath_{up} and a lower threshold Ath_{low}. They establish the Prediction Interval (PI) of the observations, which is defined in the same way as is usually performed in the bibliography [MWH97], hence assuming the following expressions:

$$Ath_{up} = p_0 + K \times \sqrt{var(E_t)} \tag{6.2}$$

$$Ath_{low} = p_0 - K \times \sqrt{var(E_t)} \tag{6.3}$$

where E_t is the prediction error in t and p_0 is the prediction of the last observation. The prediction error is given by the absolute value on the difference between the forecast and

Table 6.2: Battery of forecasting algorithms.

Method	Type
Cumulative Moving Average (CMA) [Men15]	Moving Average
Simple Moving Average (SMA) [Mul94a]	Moving Average
Double Moving Average (DMA) [Mul94a]	Moving Average
Weighted Moving Average (WMA) [FL14]	Moving Average
Simple Exponential Smoothing (EMA) [ASMW15]	Moving Average
Double Exponential Moving Average (DEMA) [Mul94b]	Moving Average
Triple Exponential Moving Average (TEMA) [Mul94a]	Moving Average
Simple Exponential Smoothing (SES) [Bro57]	Smoothing
Double Exponential Smoothing (DES) [GJ06]	Smoothing
Triple Exponential Smoothing (TES) [Win60]	Smoothing

the t observation. The variance $Var(E_t)$ is calculated considering the prediction error at the prediction period t (i.e., the horizon of the estimation). In addition, the thresholds include a parameter K, from which use case operators can adjust the sensitivity of the upper and lower limits.

6.3.4 Pattern Recognition Implementation

The instantiation of the framework considered at the performed experimentation assumes that the use case operators provide the collection of reference samples to be taken into account throughout the pattern recognition process. This implementation embraces the Attribute-Relation File Format (ARFF), Comma-Separated Values (CSV) or Packet Capture (pacp) feature descriptions in order to represent the reference datasets required for the construction of data mining models [Acw], in this way assuming their advantages, but also their drawbacks. As is the case on the Prediction component, a battery of pattern recognition and novelty detection methods is considered, which is summarized in Table 6.3. The decision of the best approach and calibration is driven by the results in terms of accuracy of a cross-validation test launched at training step.

Table 6.3: Battery of pattern recognition algorithms.

Method	Action
Decision Stump [FI92]	Classification
Reducing Error Pruning Tree [SL91]	Classification
Random Forest [Bre01]	Classification
Bootstrap Aggregation [Bre96]	Classification
Adaptive Boosting [ROM01]	Classification
Bayesian Network [BG16]	Classification
Naive Bayes [JL95]	Classification and Novelty detection
Support Vector Machines (SVM) [BL02]	Classification and Novelty detection
Generation of synthetic data + Bootstrap Aggregation [HFW08]	Novelty detection

6.3.5 Knowledge Inference Implementation

It is well known that one of the classical problems with the software that implements Rete algorithms is the lack of interoperability with other high-level languages and complex data structures, such as class hierarchies, complex knowledge representations or abstract data. Nowadays there are few open source implementations with these capabilities, as is the case of Drools [SDB]. Given that its effectiveness has been continuously proved in European projects of different nature [ArP, DC2], Drools was implemented in the current instantiation of the proposed framework in order to manage the execution of the rule-based knowledge acquisition at the knowledge inference component. As highlighted by their authors, Drools is a Business Rules Management System (BRMS) solution that provides, among others, a core Business Rules Engine (BRE) and a modification of the original Rete algorithm adapted to Object-oriented scenarios which also bring solutions to optimization problems, such as rule priorization, concurrency execution of tasks, changes on rule execution modes, synchronization of events, different forms of metadata or sliding processing.

6.4 Experiments

The following describes the evaluation scenario, reference datasets and the use cases considered throughout the performed experimentation.

6.4.1 Evaluation Scenario

Since there are not collections of 5G network traffic, the performed experimentation is focused on the study of traffic traces gathered as public domain datasets, hence facilitating the replication of the obtained results. In particular, the evaluation scenario is focused on the study of real traffic monitored on high-speed Internet backbone links published at 2016 within the CAIDA anonymized Internet Traces Dataset [DSC]. With this purpose an illustrative use case is defined, which guides the knowledge acquisition framework to infer new facts related with the variations of the traffic volume monitored. It is important to highlight that since the dataset only provides raw data, it is not possible to corroborate the incidents discovered with those identified by their authors. But such disadvantage is compensated by the fact that CAIDA is a well-known dataset with realistic information about current networks in the backbone, particularly in a data center. Throughout the experimentation, this framework has been instantiated according to the orchestration strategy introduced in [LVV17]. In this way, the analytic components act sequentially as sets of actions in the following order: pattern recognition, prediction, adaptive thresholding and knowledge inference. They are instantiated as described in Section 6.3: pattern recognition includes the battery of algorithms detailed in Table 6.3, prediction integrated the forecast methods summarized in Table 6.2, adaptive thresholding adapts the method published in [MWH97], and knowledge inference imports the engine provided by Drools [SDB]. Both prediction and pattern recognition capabilities have been evaluated according to functional standardized methodologies. Firstly, the effectiveness

of the forecast capabilities was tested adopting the M3-Competition scheme [Mak00] , in this way facilitating the comparison of the obtained results with previous publications. On the other hand, pattern recognition is validated based on the NSL-KDD dataset and the evaluation methodology proposed in [TBLG09] . As in the previous test, the results are contrasted with contributions that introduced similar features. Adaptive thresholding and knowledge inference implement well-known techniques previously considered in similar projects, so their effectiveness is assumed prior to the experimentation stage.

6.4.2 Reference Datasets

The performed experimentation applied three collections of reference data: NSL-KDD, M3-Competition and CAIDA'16. They are described below.

6.4.2.1 NSL-KDD

NSL-KDD is a dataset suggested to solve some of the inherent problems of the KDD'99 dataset, which were reviewed by Tavallaee et al. in [TBLG09], among them: presence of redundant records in the training set, record duplication, or imbalance of the number of samples per group. Note that quoting their authors "this new version of the KDD data set still suffers from some of the problems discussed by McHugh [McH00] and may not be a perfect representative of existing real networks, because of the lack of public data sets for network-based IDSs, we believe it still can be applied as an effective benchmark data set". Additionally, NSL-KDD authors analyzed the difficulty level of the samples in KDD'99, and according to the results, they proposed two different collections: $KDDTrain+\_20Percent$ KDD'99^{+}) and $KDDTest-21$ (KDD'99^{-21}), where the second includes records with difficulty level of 21 out of 21. It is important to note that according to Bhatia et al. [BSMT14], KDD'99 is one of the most referenced methodologies in the bibliography, and possibly the only one that presents a dataset of network security incidents with reliable labeling. The original KDD'99 collection was created for the competition KDD Cup, and it is based on the captures of traffic provided by the DARPA'98 dataset; in particular, legitimate (class *normal*, 97,277 (19.69%)) samples and the following simulated threats:

- Denial of Service attack (DoS): classes *back*, *land*, *neptune*, *pod*, *smurf* and *teardrop*; 391,458 (79.24%) instances.
- User to Root attack (U2R): classes *Buffer overfow*, *loadmodule*, *perl* and *rootkit*; 52 (0.01%) instances.
- Remote to Local attack (R2L); classes *Guess_passwd*, *ftp_write*, *imap*, *phf*, *multihop*, *warezmaster*, *warezclient* and *spy*; 1126 (0.23%) *instances.*
- Probing attacks: classes *satan*, *ipsweet*, *nmap* and *portsweep*; 4,107 (0.83%) instances.

Their samples are characterized by 41 different features usually divided into three groups: basic features, traffic features and content features. The first group gathers all

the attributes that can be extracted from a TCP/IP connection (e.g. duration, protocol, service, src_bytes, flag, etc.). On the other hand, the traffic features are computed with respect to a window interval, and describe host features (e.g. dst_host_count, dst_host_same_srv_rate, dst_host_serror_rate, etc.) and server features (e.g. srv_count, srv_serror_rate, diff_srv_rate, etc.). Finally, a group of features provides information able to unmask suspicious behaviors in the data portion, i.e., independent of the time period(e.g. root_shell, logged_in, hot, urgent, etc.). KDD'99 proposed as evaluation methodology to split the dataset into two groups: a 20% subset as training samples and the rest for testing. Note that NSL-KDD sanitized the original collection eliminating 78.05% of training samples (93.32% attack instances, 16.44% normal instances), and 75.15% of the test set (88.26% attack instances, 20.92% normal instances). Given that most of the discards where instances repeated in training and evaluation samples, the evaluation of classifiers with NSL-KDD displays considerably less precise results than KDD'99, posing much greater difficulty to the evaluated proposals.

6.4.2.2 M3 Competition

To the best of the author's knowledge, there are not standardized methodologies to assess the effectiveness of forecasting algorithms on 5G environments; in fact, there are also no collections of samples of these monitoring scenarios. In view of this, the most reliable way of demonstrating the capacity of the SELFNET prediction framework is to evaluate it from general purpose methodologies adapted to time series prediction. Among them it is worth considering a well-known scheme such as the M3-Competition [Mak00]. It provides a collection of 3003 time series categorized as: financial, industry, macroeconomics, microeconomics, demography and other. In order to ensure that every prediction method is able to process the proposed data, it was observed that time series have a minimum length of 14 observations for Yearly series (the median is 19 observations), 16 for Quarterly series (the median is 44 observations), 48 for Monthly time series (the median is 115 observations), and 60 for other series (the median is 63). Hence three blocks of data are clearly described: Yearly, Quarterly and Monthly. Note that all the time series are positive to avoid problems related with the various MAPE measures. If the original time series has negative values, they are replaced by zero. Table 6.4 displays the classification of these time series. In the original competition, the participants run their algorithms considering several prediction horizons (i.e., prediction periods): from $t+1$ to $t+6$ on Yearly data, from $t+1$ to $t+8$ for Quarterly data and from $t+1$ to $t+18$ for Monthly data.

The dataset was evaluated according to five metrics: symmetric Mean Absolute Percentage Error (MAPE) or sMAPE, Average Ranking, median symmetric APE, Percentage Better, and median RAE (Relative Absolute Error). Among them, the sMAPE is the most frequent in the bibliography, bearing in mind both old and recent contributions. Because of this, sMAPE is the base of the performed experimentation.

Table 6.4: Time series categories and attributes of the M3 dataset.

	Micro	Industry	Macro	Finance	Demographic	Other	Total
Year	146	102	83	58	245	11	645
Quart.	204	83	336	76	57		756
Month	474	334	312	145	111	141	1428
Other	4			29		141	174
Total	828	519	731	308	413	204	3003

6.4.2.3 CAIDA Anonymized Internet Traces 2016

The Center for Applied Internet Data Analysis (CAIDA) has published the Anonymized Internet Traces 2016 Dataset [DSC], which contains traces obtained through the passive equinix-chicago monitor located at the Equinix [PEq] datacenter in Chicago. These traces represent real internet traffic samples used for research purposes. Moreover, it is important to bear in mind that all traces are anonymized, and their payload has been removed. Thereby the resultant pcap files store only layer 3 and layer 4 packet headers to be accounted when gathering network statistics. Traces are, in fact, an hour monitored traffic captured each month. Even when traffic traces are stored each month, current yearly CAIDA datasets are a collection of four Internet traffic trace (one per quarter).A one-hour traffic trace is split in several pcap files, each of them corresponding to a one-minute traffic. Currently, CAIDA 2016 dataset has published Internet traces captured at 21 January (Ds-January), 18 February (Ds-February), 17 March (Ds-March), and 6 April (Ds-April). All of them captured from 14:00:00 to 14:59:59 h.

The first part of the experimentation was conducted using a three-minutes sample of traffic traces extracted from Ds-Jan. They correspond to network data packets captured from 14:00:00 to 14:02:59. In order to measure traffic volume, a time series of 180 elements was constructed by accumulating the total number of bytes per second. At this stage of the research, it was not feasible to extend the length of the time series due to storage and parsing time limitations. Henceforth, this sample data is referred as CAIDA'16-sample.

In the second part of the experimentation, network traffic measures were gathered from the statistics files published by CAIDA. Each statistics file corresponds to a one-minute traffic observed in a one-hour dataset. Thereby, every dataset has 60 statistics files. Unlike the described CAIDA'16-sample, there was no need to parse pcap files since every one-minute statistics file provides the total number of transmitted bytes. This is exactly the same metric used in the first part of the experimentation, being the only difference the granularity. Consequently, four time series of 60 elements were constructed from Ds-Jan, Ds-Feb, Ds-Mar and Ds-Apr, being each element of the time series the observed traffic volume expressed in bytes per minute. Henceforth, the generated time series are referred as CAIDA'16-monthly.

6.4.3 Use Case: Detection of Anomalous Traffic Volume Variations

This use case goal is to infer if the observed network traffic volume presents an anomalous behavior. For this purpose, the knowledge inference framework components are instantiated, contributing to the generation of new facts that are used to evaluate the production rules configured to infer knowledge. The process is triggered once the novelty detection capabilities of the Pattern Recognition component identify a change in the network behavior. This task requires building a model of the normal traffic behavior, which is created by assuming the first observations as reference samples and their following six attributes: Euclidean, Squared X^2, Canberra, Pearson, Bhattacharyya and Divergence distances between the last two observations. The use of these metrics in network anomaly detection is detailed reviewed in [BBK14].

Provided by the generated facts about possible anomalous traffic pattern, the Prediction component calculates the forecasting values for the time series considering 1, 5 and 10 time horizons, and the results are also inserted in the working memory. With the forecasted metrics, the Adaptive Thresholding component deduces the prediction intervals (PI) for each observation, registering them in the working memory as new acquired facts. The upper and lower thresholds are computed upon the forecasting error. Previously generated facts about abnormal traffic patterns, forecasting values and thresholds allow the Inference Engine to deduce the existence of anomalous traffic volume variations when two conditions are met: traffic volume has been labeled as abnormal and the observation is either exceeding the upper prediction interval or below the lower bound. Note that combining both of them allows considering the presence of outliers regarding the general traits of the behavior observed in the monitored environment, as well as unexpected variations from the latest observations. In this way, the incidents will be reported with greater certainty about their nature.

6.5 Results

The following describes the results obtained when analyzing the aforementioned datasets.

6.5.1 Prediction Capabilities Evaluation

The M3 dataset described in the previous section led to the evaluation of the framework under different time series; being Yearly, Monthly, Quarterly and Others the time series classifications as described in Table 6.4. The results of the evaluated forecasting methods are shown in Tables 6.5–6.8. For each method, the sMAPE value for a given forecasting horizon ($t+1$ up to $t+18$) is in fact the mean of the sMAPE values obtained for the same forecasting horizon in a set of time series (#Obs) with the same data nature.

Yearly data has been evaluated under the proposed framework and their results are detailed in Table 6.5. The obtained mean sMAPE values computed over 645 time series range from 6.6 to 9.4, thus, exposing a better accuracy for all the evaluated forecasting horizons ($t+1$ to $t+6$) compared to the other forecasting algorithms used in the M3 competition. Consequently, an average sMAPE of 7.1 computed for the 1 to 4 horizons,

and a 7.7 value for the 1 to 6 horizons, expose also an overall better accuracy in comparison with the M3 methods, which values range from 13.65 to 21.59.

Quarterly data results are shown in Table 6.6. The mean sMAPE values were computed by the proposed framework over 756 time series, and they range from 4.4 to 5.2, thus, exposing a better accuracy for most of the evaluated forecasting horizons ($t+1$ to $t+8$), being $t+1$ the only case where the framework does not show the best performance compared to the other M3 forecasting algorithms. However, the average sMAPE values of 6 for the 1 to 4, 4.9 for the 1 to 6, and 4.8 for the 1 to 8 forecasting horizons shown a better accuracy, particularly when the horizon is incremented. The average sMAPE for the existing methods are in fact ranging from 7 to 10.96 in any case.

Monthly data has also been evaluated under the framework, presenting their results in Table 6.7. The obtained mean sMAPE values computed over 1428 time series range from 9.6 to 12.7, exposing again a better accuracy for most of the evaluated forecasting horizons ($t+1$ to $t+18$) with values ranging from 9.6 to 12.7, being $t+2$ and $t+4$ the only cases where the framework has a slightly less performance of -0.5 and -0.1 for $t+2$ and $t+4$, respectively, compared with the mean SMAPE obtained by other algorithms with values ranging from 10.7 to 24.3, considering all the forecasting horizons. Hence, the average sMAPE values also show the best accuracy for the proposed framework, with values ranging from 11.1 to 11.6, being only the average sMAPE of 11.6 for the 1 to 4 horizons slightly bigger than the lowest one in this category, obtained by Theta (11.54). The remaining M3 average sMAPE values computed for the 1 to 6, 1 to 8, 1 to 12, 1 to 15 and 1 to 18 forecasting horizons range from 11.54 to 18.4 in any case. Therefore, this overall results exposed the best accuracy with Monthly data. It is worth mentioning that this set of time series are the longest used in the competition (with a mean of 115 observations).

Finally, Other data has also been evaluated following the same approach used for Quarterly data, but with 174 time series (see Table 6.8). As compared to the preceding time series categories (Yearly, Quarterly and Monthly), in this case the results were significantly better, except for the $t+1$ horizon where the mean sMAPE obtained by this proposal was 1.8 compared with the minimum value of 1.6 obtained by the Autobox 2 method. The remaining forecasting horizons shown a value ranging from 1.5 to 2.4, exposing an increasing accuracy as long as the forecasting horizon grows. In consequence, the average sMAPE values for the 1 to 4, 1 to 6 and 1 to 8 horizons show also better results when the framework performs the forecasting.

6.5.2 Pattern Recognition Capabilities Evaluation

The results obtained for the different classifiers at pattern recognition actions considering NSL-KDD'99^{+} are summarized in Table 6.9, and the results with NSL-KDD'99^{-21} are displayed in Table 6.10. On the other hand, Table 6.11 compares the effectiveness of the SELFNET Pattern Recognition set of actions with some of the most relevant proposals in the bibliography; in particular, those reviewed by Ashfaq et al. [AWH^{+}17]. This publication was released at early 2017 and discusses the effectiveness of most of the latest proposals for intrusion detection that assumed the NSL-KDD'99 evaluation methodology,

Table 6.5: SMAPE on M3-Competition for Yearly data.

Method	**Forecasting Horizon**						**Average**		**#Obs**
	$t+1$	$t+2$	$t+3$	$t+4$	$t+5$	$t+6$	**1 to 4**	**1 to 6**	
Naive	8.5	13.2	17.8	19.9	23	24.9	14.85	17.88	645
Single	8.5	13.3	17.6	19.8	22.8	24.8	14.82	17.82	645
Holt	8.3	13.7	19	22	25.2	27.3	15.77	19.27	645
Dampen	8	12.4	17	19.3	22.3	24	14.19	17.18	645
Winter	8.3	13.7	19	20	25.2	27.3	15.77	19.27	645
Comb S-H-D	7.9	12.4	16.9	24.1	22.2	23.7	14.11	17.07	645
B-J automatic	8.6	13	17.5	18.2	22.8	24.5	14.78	17.73	645
Autobox 1	10.1	15.2	20.8	22.5	28.1	31.2	17.57	21.59	645
Autobox 2	8	12.2	16.2	19	21.2	23.3	13.65	16.52	645
Autobox 3	10.7	15.1	20	20.4	25.7	28.1	17.09	20.36	645
Robust-Trend	7.6	11.8	16.6	20.3	22.1	23.5	13.75	16.78	645
ARARMA	9	13.4	17.9	19.1	23.8	25.7	15.17	18.36	645
Automat ANN	9.2	13.2	17.5	19.7	23.2	25.4	15.04	18.13	645
Flores/Pearce 1	8.4	12.5	16.9	19.1	22.2	24.2	14.22	17.21	645
Flores/Peace 2	10.3	13.6	17.6	19.7	21.9	23.9	15.31	17.84	645
PP-autocast	8	12.3	16.9	19.1	22.1	23.9	14.08	17.05	645
ForecastPro	8.3	12.2	16.8	19.3	22.2	24.1	14.15	17.14	645
SmartFcs	9.5	13	17.5	19.9	22.1	24.1	14.95	17.68	645
Theta-sm	8	12.6	17.5	20.2	13.4	25.4	14.6	17.87	645
Theta	8	12.2	16.7	19.2	21.7	23.6	14.02	16.9	645
RBF	8.2	12.1	16.4	18.3	20.8	22.7	13.75	16.42	645
ForecastX	8.6	12.4	16.1	18.2	21	22.7	13.8	16.48	645
This proposal	**6.9**	**6.6**	**7.6**	**7.2**	**8.5**	**9.4**	**7.1**	**7.7**	**645**

in this way assuming as principal classification criterion the accuracy they proved. In the case of the subset of samples NSL-KDD'99^{+}, the best classifier in SELFNET was Adaptive Boosting with 82.2% accuracy. This result is close to the best accuracy in the reviewed bibliography (84.12%), where the clustering approach introduced by Hernández-Pereira [HPSRFRAB09] was applied on flag and service features of the dataset, combined with the fuzziness based semi-supervised learning approach proposed by Ashfaq et al. Bearing in mind that in this experiment the SELFNET Pattern Recognition framework did not use data preprocessing capabilities (unlike in the aforementioned publication), it is possible to conclude that SELFNET effectiveness is sufficient for the next experiments, hence leaving preprocessing for future implementations. In the second test, the subset of samples NSL-KDD'99^{-21} was considered. The best configuration of the SELFNET Pattern Recognition framework achieved 89.9% accuracy when executed with generation of synthetic samples. The average accuracy on the latest publications is 60.3%; in particular, the best classifier tested by Ashfaq et al. demonstrated 68.2% accuracy when considering Adaptive Boosting and the previously described preprocessing. Again, it is possible consider that the achieved effectiveness is enough to validate its effectiveness.

Table 6.6: SMAPE on M3-Competition for Quarterly data.

Method	**Forecasting Horizon**							**Average**			**#Obs**
	$t+1$	$t+2$	$t+3$	$t+4$	$t+5$	$t+6$	$t+8$	**1 to 4**	**1 to 6**	**1 to 8**	
Naive	5.4	7.4	8.1	9.2	10.4	12.4	13.7	7.55	8.82	9.95	756
Single	5.3	7.2	7.8	9.2	10.2	12	13.4	7.38	8.63	9.72	756
Holt	5	6.9	8.3	10.4	11.5	13.1	15.6	7.67	9.21	10.67	756
Dampen	5.1	6.8	7.7	9.1	9.7	11.3	12.8	7.18	8.29	9.33	756
Winter	5	7.1	8.3	10.2	11.4	13.2	15.3	7.65	9.21	10.61	756
Comb S-H-D	5	6.7	7.5	8.9	9.7	11.2	12.8	7.03	8.16	9.22	756
B-J automatic	5.5	7.4	8.4	9.9	10.9	12.5	14.2	7.79	9.1	10.26	756
Autobox 1	5.4	7.3	8.7	10.4	11.6	13.7	15.7	7.95	9.52	10.96	756
Autobox 2	5.7	7.5	8.1	9.6	10.4	12.1	13.4	7.73	8.89	9.9	756
Autobox 3	5.5	7.5	8.8	10.7	11.8	13.4	15.4	8.1	9.6	10.93	756
Robust-Trend	5.7	7.7	8.2	8.9	10.5	12.2	12.7	7.63	8.86	9.79	756
ARARMA	5.7	7.7	8.6	9.8	10.6	12.2	13.5	7.96	9.09	10.12	756
Automat ANN	5.5	7.6	8.3	9.8	10.9	12.5	14.1	7.8	9.1	10.2	756
Flores/Pearce 1	5.3	7	8	9.7	10.6	12.2	13.8	7.48	8.78	9.95	756
Flores/Peace 2	6.7	8.5	9	10	10.8	12.2	13.5	8.57	9.54	10.43	756
PP-autocast	4.8	6.6	7.8	9.3	9.9	11.3	13	7.12	8.28	9.36	756
ForecastPro	4.9	6.8	7.9	9.6	10.5	11.9	13.9	7.28	8.57	9.77	756
SmartFcs	5.9	7.7	8.6	10	10.7	12.2	13.5	8.02	9.16	10.15	756
Theta-sm	7.7	8.9	9.1	9.7	10.2	11.3	12.1	8.86	9.49	10.07	756
Theta	5	6.7	7.4	8.8	9.4	10.9	12	7	8.04	8.96	756
RBF	5.7	7.4	8.3	9.3	9.9	11.4	12.6	7.69	8.67	9.57	756
ForecastX	4.8	6.7	7.7	9.2	10	11.6	13.6	7.12	8.35	9.54	756
AAM1	5.5	7.3	8.4	9.7	10.9	12.5	13.8	7.71	9.05	10.16	756
AAM2	5.5	7.3	8.4	9.9	11.1	12.7	14	7.75	9.13	10.26	756
This proposal	**5.3**	**5.2**	**4.5**	**4.7**	**4.4**	**4.8**	**4.9**	**6.0**	**4.9**	**4.8**	**756**

Table 6.7: SMAPE on M3-Competition for Monthly data.

Method	Forecasting Horizon										Average						#Obs
	$t+1$	$t+2$	$t+3$	$t+4$	$t+5$	$t+6$	$t+8$	$t+12$	$t+15$	$t+18$	1 to 4	1 to 6	1 to 8	1 to 12	1 to 15	1 to 18	
Naive	15	13.5	15.7	17	14.9	14.7	15.6	15	19.3	20.47	15.3	15.13	15.29	15.57	16.18	16.91	1428
Single	13	12.1	12.1	15.1	13.5	13.1	13.8	14.5	18.3	19.4	13.53	13.44	13.6	13.83	14.51	15.32	1428
Holt	12.2	11.6	13.4	14.6	13.6	13.3	13.7	14.8	18.8	20.2	12.95	13.11	13.33	13.77	15.51	15.36	1428
Dampen	11.9	11.4	13	14.2	12.9	12.6	13	13.9	17.5	18.9	12.63	12.67	12.85	13.1	13.77	14.59	1428
Winter	12.5	11.7	13.7	14.7	13.6	13.4	14.1	14.6	18.9	20.2	13.17	13.28	13.52	13.88	14.62	15.44	1428
Comb S-H-D	12.3	11.5	13.2	14.3	12.9	12.5	13	13.6	17.3	18.3	12.83	12.79	12.92	13.11	13.75	14.48	1428
B-J automatic	12.3	11.4	12.8	14.3	12.7	12.6	13	14.1	17.8	19.3	12.78	12.74	12.89	13.21	13.96	14.81	1428
Autobox 1	13	12.2	13	14.5	14.1	13.4	14.3	15.4	19.1	20.4	13.27	13.42	13.71	14.1	14.93	15.83	1428
Autobox 2	13.1	12.1	13.5	15.3	13.3	13.8	13.9	15.2	18.2	19.9	13.51	13.52	13.76	14.16	14.86	15.69	1428
Autobox 3	12.3	12.3	13	14.4	14.6	14.2	14.8	16.1	19.2	21.2	12.99	13.47	13.89	14.43	15.2	16.18	1428
Robust-Trend	15.3	13.8	15.5	17	15.3	15.6	17.4	17.5	22.2	24.3	15.39	15.42	15.89	16.58	17.47	18.4	1428
ARARMA	13.1	12.4	13.4	14.9	13.7	14.2	15	15.2	18.5	20.3	13.42	13.59	14	14.41	15.08	15.84	1428
Automat ANN	11.6	11.6	12	14.1	12.2	13.9	13.8	14.6	17.3	19.6	12.31	12.55	12.92	13.42	14.13	14.93	1428
Flores/Pearce 1	12.4	12.3	14.2	16.1	14.6	14	14.6	14.4	19.1	20.8	13.74	13.93	14.22	14.29	15.02	15.96	1428
Flores/Peace 2	12.6	12.1	13.7	14.7	13.2	12.9	13.4	14.4	18.2	19.9	13.26	13.21	13.33	13.53	14.31	15.17	1428
PP-autocast	12.7	11.7	13.3	14..3	13.2	13.4	14	14.3	17.7	19.6	13.02	13.11	13.37	13.72	14.36	15.15	1428
ForecastPro	11.5	10.7	11.7	12.9	11.8	12.3	12.6	13.2	16.4	18.3	11.72	11.82	12.06	12.46	13.09	13.86	1428
SmartFcs	11.6	11.2	12.2	13.6	13.1	13.7	13.5	14.9	18	19.4	12.16	12.58	12.9	13.51	14.22	15.03	1428
Theta-sm	12.6	12.9	13.2	13.7	13.4	13.3	13.7	14	16.2	18.3	13.1	13.2	13.44	13.65	14.09	14.66	1428
Theta	11.2	10.7	11.8	12.4	12.2	12.4	12.7	13.2	16.2	18.2	11.54	11.8	12.3	12.5	13.11	13.85	1428
RBF	13.7	12.3	13.7	14.3	12.3	12.8	13.5	14.1	17.3	17.8	13.49	13.18	13.4	13.67	14.21	14.77	1428
ForecastX	11.6	11.2	12.6	14	12.4	12.2	12.8	13.9	17.8	18.7	12.32	12.31	12.46	12.83	13.6	14.45	1428
AAM1	12	12.3	12.7	14.1	14	14	14.3	14.9	18	20.4	12.8	13.2	13.63	14.05	14.78	15.69	1428
AAM2	12.3	12.4	12.9	14.4	14.3	14.2	14.5	15.1	18.4	20.7	13.03	13.45	13.87	14.25	15.01	15.93	1428
This proposal	**11.0**	**11.2**	**11.7**	**12.5**	**11.6**	**11.4**	**10.6**	**9.6**	**11**	**12.7**	**11.6**	**11.6**	**11.4**	**11.1**	**11.2**	**11.4**	**1428**

Table 6.8: SMAPE on M3-Competition for other data.

Method	Forecasting Horizon							Average			#Obs
	t+1	**t+2**	**t+3**	**t+4**	**t+5**	**t+6**	**t+8**	1 to 4	**1 to 6**	**1 to 8**	
Naive	2.2	3.6	5.4	6.3	7.8	7.6	9.2	4.38	5.49	6.3	174
Single	2.1	3.6	5.4	6.3	7.8	7.6	9.2	4.36	5.48	6.29	174
Holt	1.9	2.9	3.9	4.7	5.7	5.6	7.2	3.32	4.13	4.81	174
Dampen	1.8	2.7	3.9	4.7	5.8	5.4	6.6	3.28	4.06	4.61	174
Winter	1.9	2.9	3.9	4.7	5.8	5.6	7.2	3.32	4.13	4.81	174
Comb S-H-D	1.8	2.8	4.1	4.7	5.8	5.3	6.2	3.36	4.09	4.56	174
B-J automatic	1.8	3	4.5	4.9	6.1	6.1	7.5	3.52	4.38	5.06	174
Autobox 1	2.4	3.3	4.4	4.9	5.8	5.4	6.9	3.76	4.38	4.93	174
Autobox 2	1.6	2.9	4	4.3	5.3	5.1	6.4	3.19	3.86	4.41	174
Autobox 3	1.9	3.2	4.1	4.4	5.5	5.5	7	3.39	4.09	4.71	174
Robust-Trend	1.9	2.8	3.9	4.7	5.7	5.4	6.4	3.32	4.07	4.58	174
ARARMA	1.7	2.7	4	4.4	5.5	5.1	6	3.17	3.87	4.38	174
Automat ANN	1.7	2.9	4	4.5	5.7	5.7	7.4	3.26	4.07	4.8	174
Flores/Pearce 1	2.1	3.2	4.3	5.2	6.2	5.8	7.3	3.71	4.47	5.09	174
Flores/Peace 2	2.3	2.9	4.3	5.1	6.2	5.7	6.5	3.67	7.73	4.89	174
PP-autocast	1.8	2.7	4	4.7	5.8	5.4	6.6	3.29	4.07	4.62	174
ForecastPro	1.9	3	4	4.4	5.4	5.4	6.7	3.31	4	4.6	174
SmartFcs	2.5	3.3	4.3	4.7	5.8	5.5	6.7	3.68	4.33	4.86	174
Theta-sm	2.3	3.2	4.3	4.8	6	5.6	6.9	3.66	4.37	4.93	174
Theta	1.8	2.7	3.8	4.5	5.6	5.2	6.1	3.2	3.93	4.41	174
RBF	2.7	3.8	5.2	5.8	6.9	6.3	7.3	4.38	5.12	5.6	174
ForecastX	2.1	3.1	4.1	4.4	5.6	5.4	6.5	3.42	4.1	4.64	174
This proposal	**1.8**	**2.3**	**2.2**	**2.0**	**2.3**	**1.5**	**2.4**	**2.1**	**2.0**	**2.0**	**174**

Table 6.9: Results when analyzing NSL-KDD'99^{+}.

Classifier	Class	TPR	FPR	Precision	Recall	F-Measure	MCC	AUC	PRC Area	Accuracy
Decision Stump	Normal	0.955	0.731	0.695	0.955	0.804	0.642	0.819	0.683	0.799
	Anomaly	0.683	0.045	0.952	0.683	0.795	0.642	0.819	0.831	
	Average	0.8	0.162	0.841	0.8	0.8	0.642	0.819	0.767	
RepTree	Normal	0.909	0.256	0.729	0.909	0.809	0.649	0.822	0.721	0.815
	Anomaly	0.744	0.091	0.915	0.744	0.821	0.649	0.822	0.858	
	Average	0.815	0.162	0.835	0.815	0.816	0.649	0.822	0.799	
Random Forest	Normal	0.973	0.323	0.695	0.973	0.811	0.658	0.959	0.947	0.803
	Anomaly	0.677	0.027	0.971	0.677	0.798	0.658	0.959	0.961	
	Average	0.804	0.155	0.852	0.804	0.803	0.658	0.959	0.955	
Bootstrap Aggregation	Normal	0.917	0.249	0.736	0.917	0.816	0.663	0.928	0.909	0.822
	Anomaly	0.751	0.083	0.923	0.751	0.828	0.663	0.928	0.916	
	Average	0.822	0.155	0.842	0.822	0.823	0.663	0.928	0.913	
Adaptive Boosting	Normal	0.968	0.399	0.648	0.968	0.776	0.589	0.935	0.919	0.822
	Anomaly	0.601	0.032	0.961	0.601	0.74	0.589	0.935	0.941	
	Average	0.759	0.19	0.826	0.759	0.755	0.589	0.935	0.932	
Bayesian Network	Normal	0.973	0.429	0.632	0.973	0.766	0.57	0.945	0.94	0.759
	Anomaly	0.571	0.027	0.965	0.571	0.718	0.57	0.945	0.955	
	Average	0.744	0.2	0.822	0.744	0.739	0.57	0.945	0.949	
Naive Bayes	Normal	0.931	0.367	0.657	0.931	0.771	0.572	0.895	0.844	0.761
	Anomaly	0.633	0.69	0.924	0.633	0.751	0.572	0.914	0.911	
	Average	0.761	0.198	0.809	0.761	0.759	0.572	0.908	0.882	
SVM	Normal	0.954	0.355	0.670	0.954	0.787	0.608	0.799	0.659	0.77
	Anomaly	0.645	0.046	0.948	0.645	0.768	0.608	0.799	0.814	
	Average	0.778	0.179	0.829	0.778	0.776	0.608	0.799	0.747	
Synthetic data	Normal	0.922	0.302	0.698	0.922	0.794	0.620	0.916	0.901	0.794
	Anomaly	0.698	0.078	0.922	0.698	0.794	0.620	0.918	0.913	
	Average	0.794	0.175	0.825	0.794	0.794	0.620	0.917	0.908	

Table 6.10: Results when analyzing NSL-KDD'99[−21].

Classifier	Class	TPR	FPR	Precision	Recall	F-Measure	MCC	AUC	PRC Area	Accuracy
Decision Stump	Normal	0.848	0.416	0.311	0.848	0.456	0.33	0.716	0.292	0.631
	Anomaly	0.584	0.152	0.945	0.584	0.722	0.33	0.716	0.893	
	Average	0.632	0.2	0.83	0.632	0.674	0.33	0.716	0.783	
RepTree	Normal	0.635	0.342	0.292	0.963	0.4	0.231	0.751	0.372	0.643
	Anomaly	0.658	0.365	0.89	0.658	0.757	0.231	0.751	0.923	
	Average	0.654	0.361	0.782	0.654	0.692	0.231	0.751	0.823	
Random Forest	Normal	0.875	0.425	0.314	0.875	0.462	0.347	0.794	0.576	0.629
	Anomaly	0.575	0.125	0.954	0.575	0.718	0.347	0.794	0.935	
	Average	0.63	0.179	0.838	0.63	0.671	0.347	0.794	0.87	
Bootstrap Aggregation	Normal	0.637	0.35	0.281	0.637	0.396	0.225	0.743	0.465	0.647
	Anomaly	0.65	0.363	0.89	0.65	0.751	0.225	0.743	0.922	
	Average	0.647	0.361	0.78	0.647	0.687	0.225	0.743	0.839	
Adaptive Boosting	Normal	0.866	0.518	0.217	0.866	0.413	0.272	0.724	0.394	0.522
	Anomaly	0.482	0.134	0.942	0.482	0.638	0.272	0.724	0.901	
	Average	0.552	0.204	0.82	0.552	0.597	0.272	0.724	0.809	
Bayesian Network	Normal	0.878	0.563	0.257	0.878	0.398	0.25	0.744	0.486	0.516
	Anomaly	0.437	0.122	0.942	0.437	0.597	0.25	0.744	0.928	
	Average	0.517	0.202	0.817	0.517	0.561	0.25	0.744	0.848	
Naive Bayes	Normal	0.678	0.469	0.243	0.678	0.358	0.161	0.648	0.294	0.557
	Anomaly	0.531	0.322	0.882	0.531	0.663	0.161	0.65	0.876	
	Average	0.558	0.348	0.766	0.558	0.607	0.161	0.65	0.77	
SVM	Normal	0.180	0.001	0.982	0.180	0.304	0.385	0.589	0.325	0.850
	Anomaly	0.999	0.820	0.846	0.999	0.916	0.385	0.589	0.846	
	Average	0.851	0.672	0.871	0.851	0.805	0.385	0.589	0.752	
Synthetic data	Normal	0.905	0	1	0.095	0.905	N/A	N/A	N/A	0.899
	Anomaly	0	0.095	0	0	0	N/A	N/A	N/A	
	Average	0.905	0	1	0.095	0.95	N/A	N/A	N/A	

Table 6.11: Comparison with related works in terms of accuracy.

Method	**NSL-KDD'99^{+}(%)**	**NSL-KDD'99^{-21}(%)**
J48	81.05	63.97
Naive Bayes	76.56	55.77
NB tree	82.02	66.16
Random forests	80.67	63.25
Random tree	81.59	58.51
M-L perceptron	77.41	57.34
SVM	69.52	42.29
Fuzzy	82.41	67.06
Fuzzy D&D	84.12	68.82
This proposal (Classification)	**82.2**	**64.7**

6.5.3 Use Case Evaluation

The following sections describe the two experiments carried on upon the CAIDA'16 reference dataset, analyzed under different levels of data granularity for each: per second and per minute.

6.5.3.1 Experiment 1: CAIDA'16-Sample

The first step on the CAIDA traffic volume analysis according to the aforementioned use case is novelty detection. With this purpose, the first 35 observations on the monitored environment are considered as reference samples for building the normal network usage model. The evaluation of the model demonstrated 91.4894% accuracy when tested via cross-validation. The best selected pattern recognition setting was the combination of generating synthetic data as counterexample [HFW08] and its analysis with Bootstrap Aggregation [Bre96] based on decision stump [FI92]. Discordant traffic volume values were monitored at observations 86–88 (21 January 2016 14:01:25 to 14:01:29), 113 (21 January 2016 14:01:54), 115 (21 January 2016 14:01:56), 139–141 (21 January 2016 14:02:20 to 14:02:23). Figure 6.2 summarizes the anomalous observations discovered. The impact of the six attributes taken into account is illustrated in Figure 6.3. As can be observed, each of them highlights the fact that at the aforementioned observations on the traffic volume, there is a discordant with the reference data.

The next knowledge acquisition step is to infer new facts from predictions. The obtained results are summarized in Table 6.12, and Figure 6.4 illustrates the evolution of the predictions for horizon 1 (Figure 6.4a), horizon 5 (Figure 6.4c) and horizon 10 (Figure 6.4e). From them it is easy deduce that the higher horizon, the higher forecast error. On the other hand, their different adaptive thresholds are shown in Figure 6.4b,d,f. The thresholds provide greater margin of error when the forecasting error is higher. Because of this, the selection of an appropriate horizon plays an essential role in the use case effectiveness, since it conditions the level of restriction on which operates the knowledge acquisition framework.

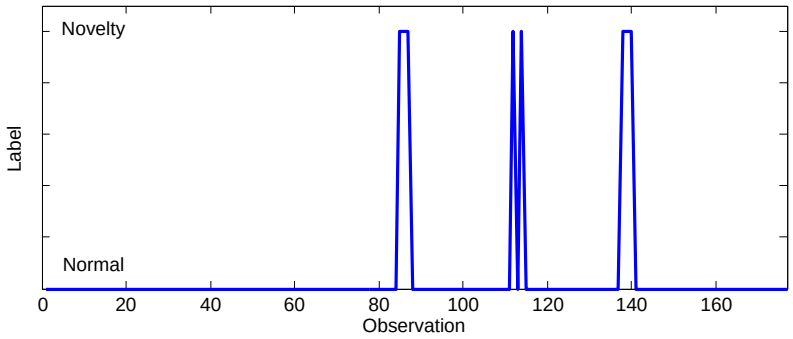
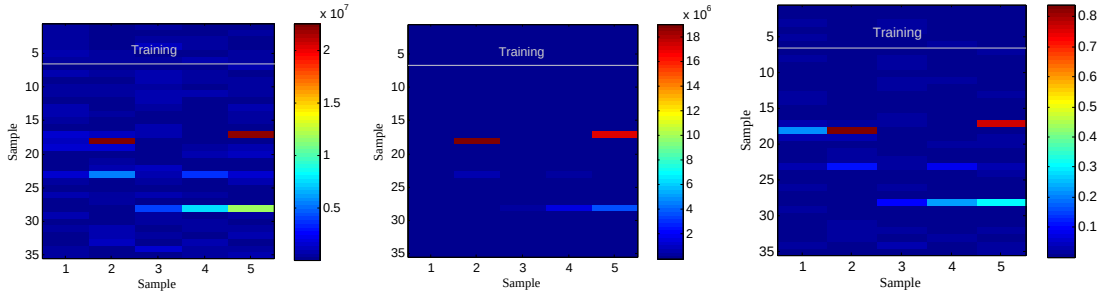


Figure 6.2: Discordant observations at novelty detection for CAIDA'16-sample.



(a) Euclidean (b) Quadratic X^2 (c) Canberra

(d) Pearson (e) Bhattacharyya (f) Divergence

Figure 6.3: Metric variations on samples.

Another aspect to keep in mind is the impact of the K adjustment value at the decisions taken. This parameter regulates the restraint of the adaptive thresholds. Figure 6.5 illustrated the variation of the ratio of observations tagged as normal when modifying K. Regardless of the prediction horizon, when K shows lower values the number of observations labeled as unexpected is higher; hence the level of restriction on which the framework operates is higher. Conversely, as K grows the normal labeling rate increases, in this way overlooking situations that in the previous cases were considered discordant.

Finally, letting a forecasting horizon of 1 observation and $K = 1$, Figure 6.6a illustrated the traffic volume evolution on CAIDA'16-sample and the adaptive thresholds inferred at the proposed framework. Figure 6.6b summarizes the unexpected observations discovered, which provide the rest of the information required to produce conclusions (symptoms). Unexpected traffic volumes occur at observations 86–93 (21 January 2016 14:01:25 to

14:01:34), 113 (21 January2016 14:01:54), 115 (21 January 2016 14:01:56), 139–143 (21 January 2016 14:02:20 to 14:02:25) and 146 (21 January 2016 14:02:25). By contrasting Figures 6.2 and 6.6b it is possible deduce when the knowledge acquisition instantiation of the proposed framework infers symptoms as final facts related with the implemented use case (i.e., observations tagged as "suspicious"). It takes place each time a novelty behavior is discovered and the traffic volume is unexpected, which occurs at observations 86–88 (21 January 2016 14:01:25 to 14:01:29), 113 (21 January 2016 14:01:54), 115 (21 January 2016 14:01:56), 139–141 (21 January 2016 14:02:20 to 14:02:23).Suspicious variations on the volume of the monitored data are reported to the decision-making sub-layer, where the countermeasures to be deployed are planned and orchestrated.

Table 6.12: Forecasting results for each horizon.

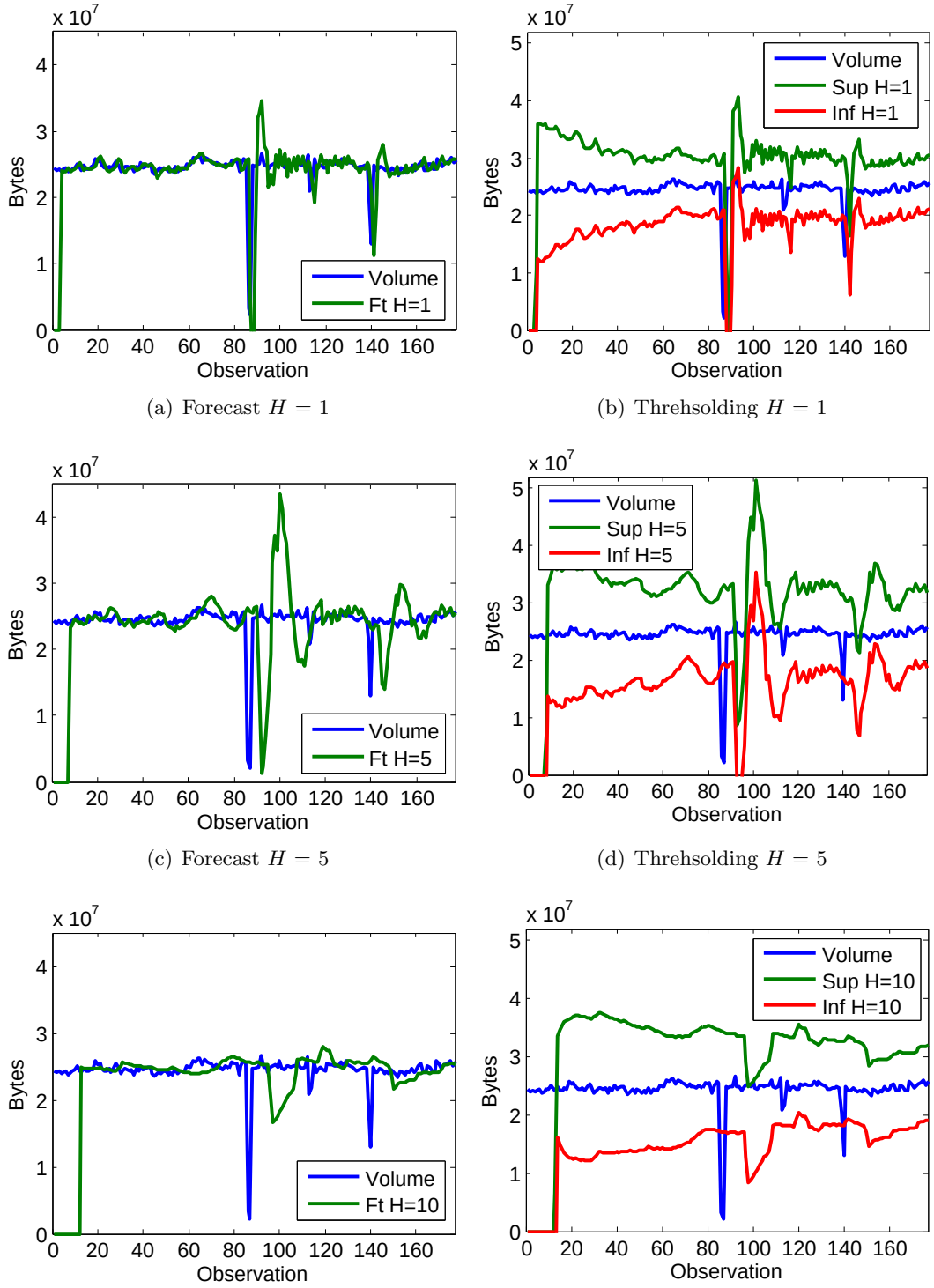
Forecasting Horizon (H)	Selected Algorithm	Parameter Calibration	SMPAPE
1	Multiplicative Holt-Winters	alpha = 0.5, beta = 0.1, gamma = 0.9	0.0004
5	Multiplicative Holt-Winters	alpha = 0.1, beta = 0.3, gamma = 0.9	0.6972
10	Additive Holt-Winters	alpha = 0.1, beta = 0.3, gamma = 0.1	1.7622

6.5.3.2 Experiment 2: CAIDA'16-monthly

This experiment was performed upon CAIDA'16-monthly data, by following the same approach conducted in the previous section. Since CAIDA'16-monthly is composed by four time series (traffic traces collected at January, February, March and April), they are individually analyzed by the framework. Being novelty detection and forecasting the set of actions in the knowledge acquisition process, the conclusions deduced by the framework are summarized in Figure 6.7. Observations are shown in Figure 6.7a,c,e,g; and the comparison between novelty detection and unexpected traffic detected at each monthly dataset are plotted in Figure 6.7b,d,f,h, which are the required facts to produce symptoms.

When analyzing CAIDA'16 at January and February it can be concluded that even when discordant observations are detected in the novelty detection stage, unexpected traffic volumes are not observed in any of the time series. Thereby, there are no anomalous traffic volume variations reported by the framework. It is explained due to the required conditions (novelty and unexpected traffic) do not occur simultaneously at any observation.

On the other hand, March and April CAIDA'16 datasets detected both unexpected and discordant traffic volume observations in the analyzed time series. By contrasting the traffic labeled as "unexpected" with the novelty detection results at the March dataset (Figure 6.7f), the inference of anomalous traffic symptoms take place at observations 29 (17 March 2016 14:28) and 31 (17 March 2016 14:30). Likewise, at the April dataset (Figure 6.7h), symptoms are inferred at observations 29 (6 April 2016 14:28), 31 (6 April 2016 14:30), and 32 (6 April 2016 14:31); where network traffic is simultaneously considered "unexpected" and "fluctuant", so the two required conditions to trigger the "suspicious" traffic symptoms are met.



(a) Forecast $H = 1$

(b) Threhsolding $H = 1$

(c) Forecast $H = 5$

(d) Threhsolding $H = 5$

(e) Forecast $H = 10$

(f) Threhsolding $H = 10$

Figure 6.4: Evolution of prediction and adaptive thresholding on CAIDA'16 sample.

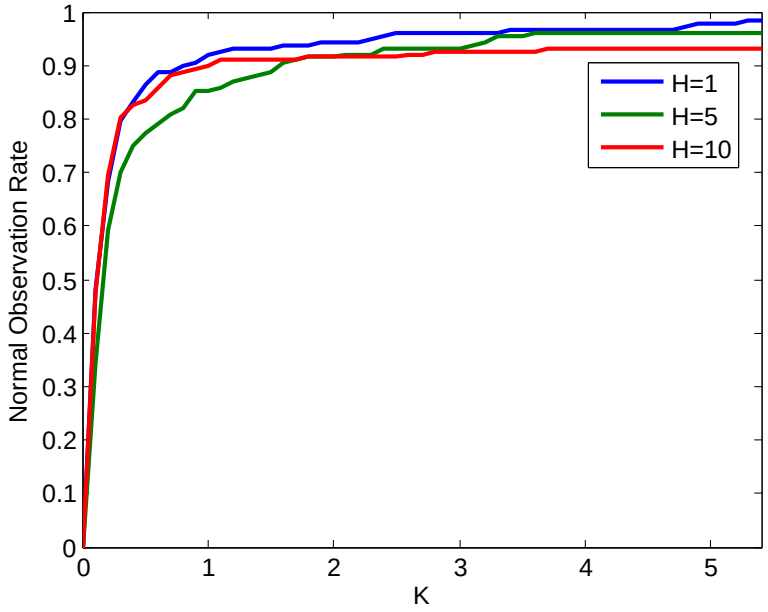
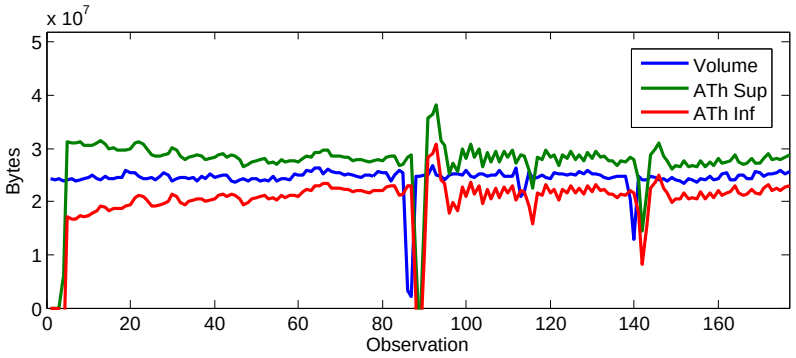
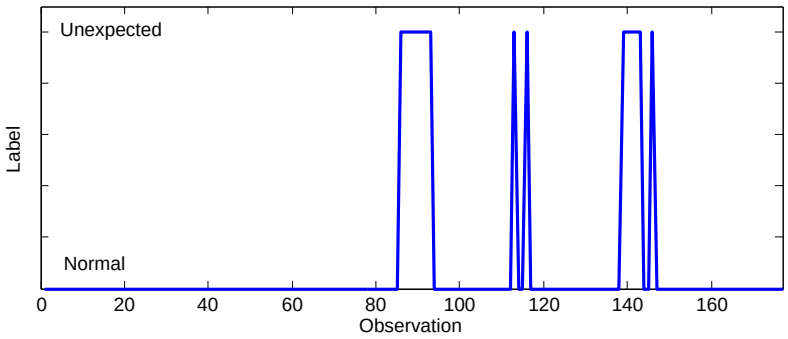


Figure 6.5: Normal observation rate when varying K.

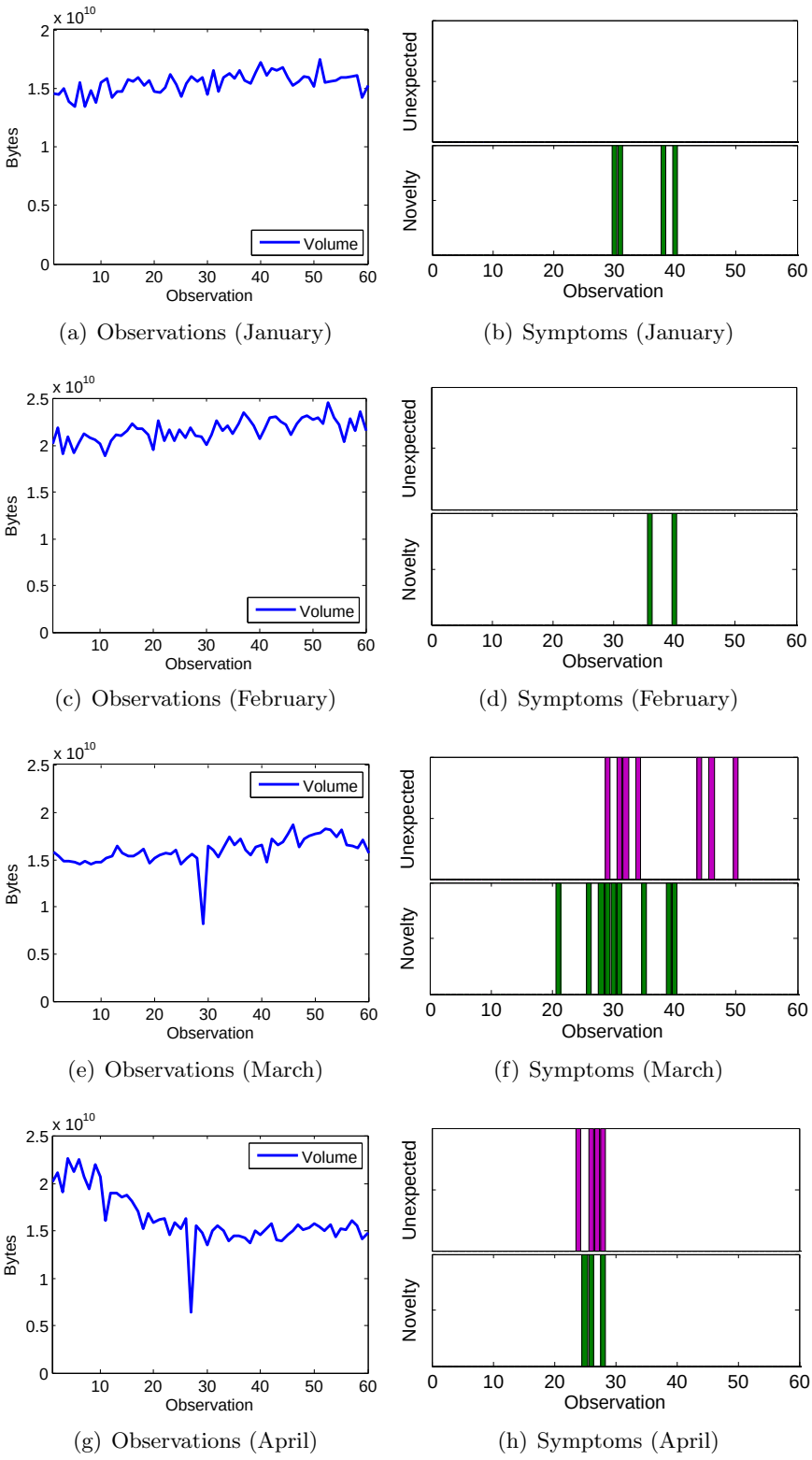


(a) Traffic volume variation on CAIDA'16-sample



(b) Unexpected observations on CAIDA'16-sample

Figure 6.6: Thresholds and Unexpected traffic on CAIDA'16-sample.



(a) Observations (January) (b) Symptoms (January)

(c) Observations (February) (d) Symptoms (February)

(e) Observations (March) (f) Symptoms (March)

(g) Observations (April) (h) Symptoms (April)

Figure 6.7: Evolution of observations, thresholding and novelty detection on CAIDA'16 monthly.

6.6 Final Remarks

This chapter presented a reasoning-based framework for the acquisition of knowledge in 5G networks, and an instantiation of this framework has been performed. For this purpose, every component has been equipped with very basic capabilities, which has led to the implementation of well-known data mining algorithms and machine learning schemes within each of them. The experimentation has focused on two fundamental aspects: demonstrating that each component operates properly and testing the potential of the proposal in a real use case. At the first stage, the functional standards NSL-KDD and M3-competition have been considered. The obtained results corroborated the efficacy of the deployed components by comparison with similar proposals. On the other hand, the defined use case allowed the acquisition of knowledge related to variations of traffic volume on the monitoring environment. This has been tested with real traffic, in particular with traces provided by the CAIDA'16 collection. The results effectively demonstrated its accuracy when generating useful information for the management of the occurred incidents. In this way, it contributes on the provision of self-management capabilities adapted to emergent network contexts.

Chapter 7

Entropy-based Economic Denial of Sustainability (EdoS) detection

In recent years, an important increase in the amount and impact of Distributed Denial of Service (DDoS) threats has been reported by different information security organizations. They typically target on the depletion of the computational resources of the victim system to prevent their operability. Inspired by such methods, Economic Denial of Sustainability (EDoS) attacks pose a similar motivation, but adapted to cloud computing environments, where the denial is targeted on damaging the economy of both suppliers and customers. Therefore, the most common EDoS approach is making the offered services unsustainable by exploiting their auto-scaling algorithms. In order to contribute to their mitigation, this chapter introduces a novel EDoS detection method based on the study of entropy variations related with metrics considered to infer the behavior of the monitored environment. Through the prediction and definition of adaptive thresholds, unexpected behaviors capable of fraudulently demand new resources are distinguished. For validating purposes, an experimental scenario adapted to the singularities of the EDoS threats has been implemented and the proposal accuracy has been effectively assessed.

With the purpose of cooperate with the research community towards their mitigation, the following main contributions are accomplished:

- A multi-layered architecture for EDoS attack detection, which describes the management of the acquired information from its monitoring to the notification of possible threats.
- A novel entropy-based EDoS detection approach, which assuming its original definition, allows to discover unexpected behavior on local-level metrics related with the auto-scaling capabilities of the victim system.
- An evaluation methodology adapted to the singularities of the EDoS threats and the assumptions driven by their original definition.
- Comprehensive experimental studies that validate the proposed detection strategy, in this way motivating its adaptation to future use cases.

In order to facilitate the understanding of this chapter, their contents are structured into seven sections. Section 7.1 summarizes the assumptions and limitations considered in the EDoS detection architecture. Section 7.2 describes the EDoS detection architecture proposed in this work. The performed experimentation is described in Section 7.3, and the obtained results are then discussed in Section 7.4. Finally, Section 7.5 presents the conclusions of this chapter.

7.1 Assumptions and limitations

With the purpose of establish the basis for defining an appropriate design methodology, the peculiarities of the conventional Denial of Service attacks, the legitimate mass access to the protected services (i.e., flash crowds), and their differences with the Denial of Sustainability threats have been taken into account. They allowed to define the following assumptions and limitations concerning the proposal described in the rest of this section:

- As remarked by Hoff in the original definition of EDoS attacks [Hof08], they pose threats that do not aim on deny the service of the victim systems, but increase the economic cost of the services they offer to make them unsustainable.
- Hereinafter, Chris H. clarified that at network-level, EDoS threats resemble activities performed by legitimate users [Hof09]. This implies that the distribution of the different network metrics (number of request, number of sessions, frequency, bandwidth computation, etc.) does not vary significantly when these attacks are launched. This is because in order to ensure their effectiveness, they must go unnoticed.
- It is possible to identify EDoS attacks by analyzing performance metrics at local-level. Given that at network-level there are no differences between EDoS and normal traffic, the requests performed by these threats must involve a greater operational cost.
- Requests performed by EDoS attacks have a similar quality to those from legitimate users (for example, a similar success rate). However, attackers may exploit vulnerabilities (usually at Application layer) to extend their impact [SGSC16].
- DDoS attacks usually originate from a large number of clients, where each of them performs a huge number of low-quality requests. On the other hand, EDoS attacks also come from many sources, but each client performs an amount of requests similar to that of legitimate users. Unlike in flash crowds, EDoS attacks affect the predictability of the performance metrics related to the costs resulting from attending the requests served by the victim [ZJW+14].
- The security mechanisms implemented on each software component involved in the enforcement of cloud auto-scaling policies should raise the likelihood of preventing EDoS attacks. However, such isolated security approach lacks a global detection strategy for disclosing unnoticed EDoS patterns. Hence, this proposal delves into the

EDoS detection analytics methods, disregarding the security considerations assumed on the development and implementation of the different components.

7.2 EDoS Detection Architecture

Based on the assumptions and limitations stated so far, it is possible to assume that, by studying the predictability of performance metrics at local-level (e.g., processing time, memory consumption, input and output operations, CPU consumption, etc.), it is possible successfully identify EDoS attacks. This is taken into account in the following subsections, where the introduced detection strategy is described. The proposal has the architecture illustrated in Figure 7.1. Therefore, it must perform three main tasks: (1) monitoring and aggregation; (2) novelty detection and (3) decision-making. They are described below.

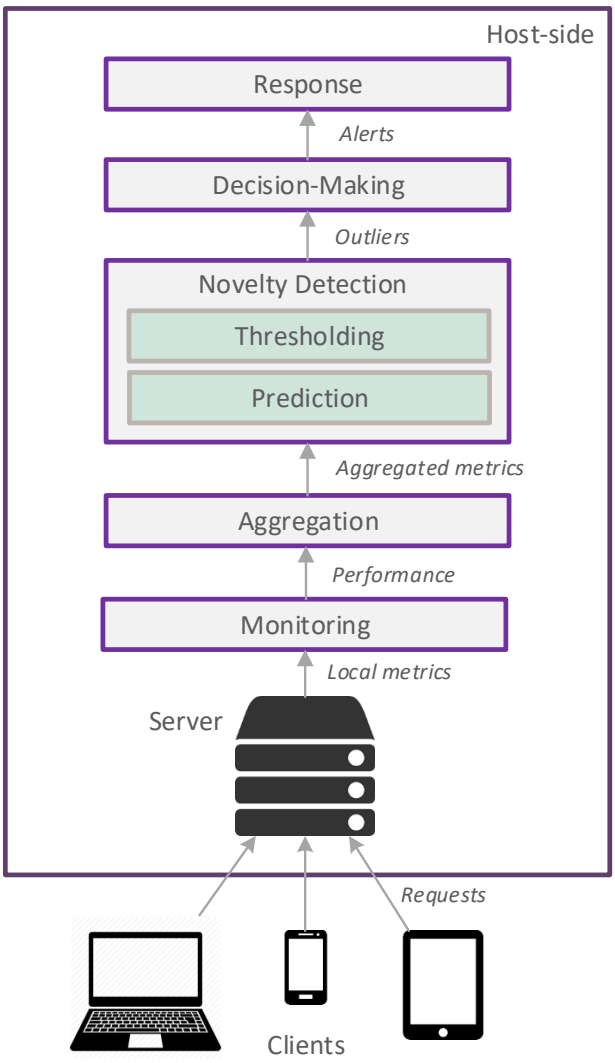


Figure 7.1: Architecture for EDoS attack detection.

7.2.1 Monitoring and Aggregation

At the monitoring stage, the factual knowledge necessary to deduce the nature of the requests to be analyzed is collected. Therefore, the detection system monitors local metrics related to the operational cost of responding the received request. Assuming that in order to success, EDoS attacks attempt to trigger the auto-scaling mechanisms of the victim-side, the metrics that determine these actions acquire special relevance. Note that they are widely studied in the bibliography, which vary according to the management services. Examples of well-known local-level metrics are: CPU utilization, warming time, response time, number of I/O requests, bandwidth or memory consumption [BBBS17, SGSC16]. Because of its relevance in the recent Cloud computing commercial solutions (e.g., Google Cloud, Amazon EC2, etc.) the performed experimentation considered the percentage CPU usage of the victim system.

On the other hand, it is important to borne in mind that the analysis of the predictability degree of events has played an essential role in the defense against conventional DDoS threats. Among the most used aggregated metrics, it is worth mentioning the classical entropy adaptation to the information theory proposed by Shannon [Sha48]. Note that in approaches like [BBK15] it is demonstrated its effectiveness when applied to DDoS detection, being a strong element in the discovery of flooding threats. Recent publications such as [BS15, IT11, JSTD16] tried to adapt this paradigm to the EDoS problem. However, most of them made the mistake of only considering information monitored at network-level, hence ignoring part of the information that truly defines the auto-scaling policies. Because of this, the Aggregation stage of the proposed method calculates the information entropy $H(X)$ of the $\{x_1, x_2, \ldots, x_n\}$ instances of the qualitative variable X monitored per observation, as well as their $\{p_1, p_2, \ldots, p_n\}$ probabilities. The proposed detection scheme defines X as "the response time (rate) to the different requests performed by each client". Given that X describes discrete events, its entropy is expressed as follows:

$$H(X) = \sum_{i=1}^{n} p_i \log_a p_i \tag{7.1}$$

where $\log_a b . \log_b x = \log_a x$. $H(X)$ is normalized, hence being calculated when dividing the obtained value by the maximum observable entropy $\log_b n$. When the maximum entropy is reached, all the monitored clients made requests with the same CPU overload; on the contrary, if the registered entropy is 0 then 1) a single customer carried out all the requests, or 2) there was no CPU consumption during the observation period. The sequence of monitored entropies is studied as a time series $H(X)_{t=0}^{N}$.

7.2.2 Novelty Detection

The next analytic step is to recognize the observations that significantly vary from normal behaviors. This is a one-class classification problem where it is assumed that the normal data compiles the previous $H(X)_{t=1}$,..., $H(X)_{t=N-1}$ observations and it is intended to deduce if $H(X)_{t=N}$ belongs to the same activities. The bibliography provides

a large variety of solutions to this problem [PCCT14]. However, because it was assumed that EDoS attacks could be identified by discovering discordances at the predictability of local-level aggregated metrics [ZJW+14], the proposed system implements a forecasting approach.

7.2.2.1 Detection Criteria

In particular, the entropy for certain horizon h, $\hat{H}(X)_{t=N+h}$, is predicted. Hence, letting the following Euclidean distance:

$$dist(o, \hat{o}) = \sqrt{(\hat{H}(X)_{t=N+h} - (X)_{t=N+h})^2} \tag{7.2}$$

If $(X)_{t=N+h}$ differs from $\hat{H}(X)_{t=N+h}$, so $dist(o, \hat{o}) > 0$ an unexpected behavior is detected. The significance of this anomaly is established by two adaptive thresholds: Upper Threshold (Th_{sup}) and Lower Threshold (Th_{inf}). A novelty was discovered if any of the following conditions is met:

$$\begin{aligned} dist(o, \hat{o}) > 0 \quad and \quad H(X)_{t=N+h} > Th_{sup} \\ dist(o, \hat{o}) > 0 \quad and \quad H(X)_{t=N+h} < Th_{inf} \end{aligned} \tag{7.3}$$

7.2.2.2 Prediction

The implemented prediction methodology adopted the Autoregressive Integrated Moving Average $ARIMA(p, d, q)$ paradigm [HT82], which defined by the following general-purpose forecast model:

$$Y_{\tau-1} - a_1 Y_{\tau-1} - \cdots - a_{p'} Y_{\tau-p'} = \epsilon_t + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_{t-q} \tag{7.4}$$

where a_i are the parameters of the autoregressive part, θ_i are the parameters of the moving average part and ϵ_t is white noise. The adjustment of p, d, q may be the ARIMA model equal to other classical forecasting models. For example simple random walk ($ARIMA(1, 1, 0)$), $AR(ARIMA(1, 0, 0))$, $MA(ARIMA(0, 0, 1))$, simple exponential smoothing ($ARIMA(0, 1, 1)$), double exponential smoothing ($ARIMA(0, 2, 2)$), etc. Predictions ($\hat{y}_t$) on ARIMA models are inferred by a generalization of the autoregressive forecasting method expressed as follows:

$$\hat{y}_t = \mu + \phi_1 Y_{\tau-1} + \phi_p Y_{\tau-p} - \phi_1 \epsilon_{t-1} - \cdots - \phi_q \epsilon_{t-q} \tag{7.5}$$

and the calibration of the adjustment parameters p, d, q considered the Akaike Information Criterion (AIC) as described in [OHT05].

7.2.2.3 Adaptive Thresholding

On the other hand, the adaptive thresholds define the Prediction Interval (PI) of the sensor, which is deduced in the same way as it is usually described in the bibliography, hence assuming the following expressions:

$$\begin{aligned} Th_{sup} &= H(X)_{t=N+h} + K\sqrt{\sigma^2 var(dist(o,\hat{o}))} \\ Th_{inf} &= H(X)_{t=N+h} - K\sqrt{\sigma^2 var(dist(o,\hat{o}))} \end{aligned} \tag{7.6}$$

and being K the confidence interval of the estimation (by default $Z_{\frac{\alpha}{2}}$). Note that despite linking its value to the normal distribution, it was demonstrated that when time series does not approach such distribution, the obtained error is unrepresentative [HKOS05]. Figure 7.2 illustrates an example of novelty detection. In the first 60 observations non $H(X)$ exceeds the adaptive thresholds; but at observation 61 an EDoS attack was launch, and the inferred changes meet the conditions to be considered novel.

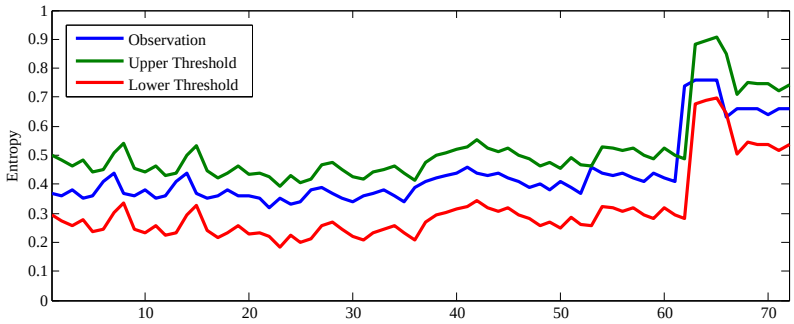


Figure 7.2: Example of novelty detection.

7.2.3 Decision-Making and Response

According to the principles of anomaly-based intrusion detection compiled and discussed by Chandola et al. [Cha09], once assumed the appropriate premises, the identification of discordant behaviors may be indicative of malicious activities. As stated at the beginning of this chapter, the introduced EDoS detection system lies on the original definitions of C. Hoff and R. Cohen. Therefore, when a local metric directly related with triggering auto-scaling capabilities on Cloud computing became unpredictable, it is possible deduce that the protected environment is misused, hence jeopardized. This occurs when $dist(o,\hat{o}) > 0$ and 1) $H(X)_{t=N+h} > Th_{sup}$ or 2) $H(X)_{t=N+h} < Th_{inf}$. Because the performed research focused only on detect the threats, its response is to notify the detected incident. The report may trigger mitigation measures such as initiate more restrictive control access [MARH13, KN09, AAB13] or deploy source identification capabilities [ITJ12] (which decision and development is out of scope). Therefore, it entails a good complement to many of the proposals in the bibliography.

7.3 Experiments

The following sections describe the Cloud-based testbed and related architectural components considered throughout the performed experimentation. They are depicted in Figure 7.3.

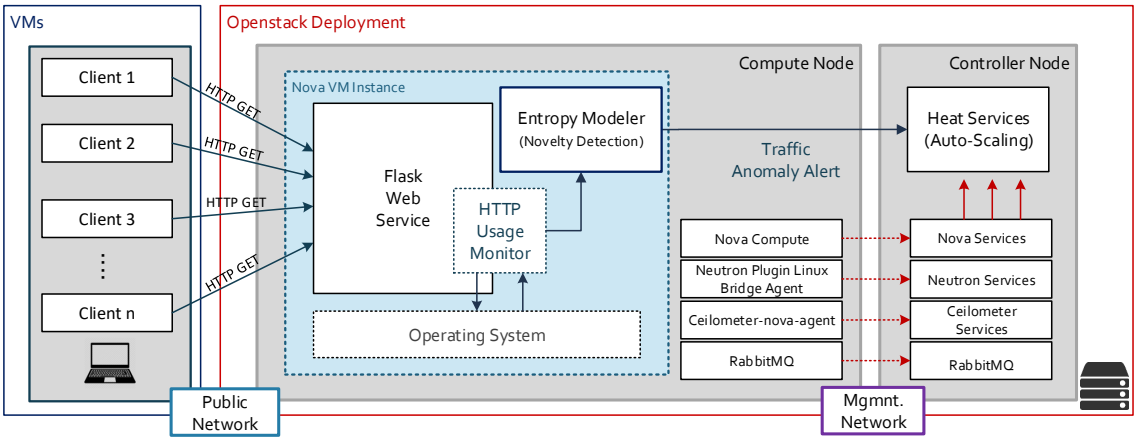


Figure 7.3: Cloud execution environment for experiments.

7.3.1 Execution Environment

The experimental cloud computing environment was built with Openstack [Ope], a well-known open source cloud platform suitable to deploy public and private cloud environments of any size. The auto-scaling features of this cloud platform have also been tested effectively on recent publications. The Openstack deployment for the experimental testbed was composed by one controller node and one compute node. The controller runs core Openstack services and it also holds the Networking (Neutron), Compute (Nova) essentials, Telemetry (Ceilometer) and Message Queue (RabbitMQ) services. In addition, it runs the Orchestration (Heat) services to allow the configuration of auto-scaling policies. The compute node runs in a separate server, hosting the Nova core services. A new Compute instance has been launched to deploy the web service used for experimentation. This virtual instance runs an Ubuntu 16.04-x64 server with 8 CPU cores and 8 GB of RAM memory.

On top of the operating system, a REST (Representational State Transfer) web service written in Flask [SFl] has been implemented. A REST web service has been chosen due to its simplicity and rapid development. REST is the predominant web API design model built upon HTTP methods, which accommodates the system to interact with several entities (i.e., humans, IoT devices). In REST every client request 1) only generates a single server response (one-shot) and 2) every response must be generated immediately (one-way). This request-response model is suitable to focus the analysis on the measurement of CPU processing times, by tracking the connected user and the impact of its client requests on the CPU consumption.

In addition to the web service, two modules were developed to be run in the background: The HTTP Usage Monitor module and the Entropy Modeler. The former logs information regarding the monitoring of client requests processing times, whereas the latter performs novelty detection methods to trigger anomaly-based alerts to the Openstack orchestration services.

On the client-side, a set of REST-clients have been deployed to generate traffic according to several execution scenarios. The implementation details and characteristics

of the components tested in the experimentation stage are explained in the forthcoming sections.

7.3.2 Server-Side Components

The following describes the deployed server-side components: RESTful Web Service, HTTP Usage Monitor and the Entropy Modeler.

7.3.2.1 RESTful Web Service

To facilitate a seamless interaction with HTTP clients, a REST web service has been implemented on Flask, a Python-based framework for rapid development of web applications. The REST service exposes four HTTP endpoints that produce the execution of different list-sorting operations on the server, each of them consumes a different amount of CPU time which is measured in the background. The endpoints and their average execution times are summarized in Table 7.1.

Table 7.1: HTTP GET endpoints and CPU average cost.

URI	Parameters	Avg. CPU Time in Sec. (1000 exec.)
/1	?id={clientID}	0.02158
/2	?id={clientID}	0.02781
/3	?id={clientID}	0.03673
/4	?id={clientID}	0.33604

7.3.2.2 HTTP Usage Monitor

Once the server receives a client HTTP request, the Usage Monitor module permanently measures the amount of CPU time consumed to process the request before sending the response back to the client. The module makes use of Python libraries and standard Linux utilities to track the CPU consumption per each client request. The collected data is then aggregated per client in configurable time intervals before being logged to the system. If more than one client connection is being observed in the given time interval, only the sum (aggregated metric) of all the processing times is logged. This allows the creation of a time series, required for the next processing level.

7.3.2.3 Entropy Modeler

This module gathers the time series logged by the HTTP Usage Monitor and computes the entropy of the CPU time usage of the different requests performed by each client. With the resultant normalized entropy, the module forecasts the next h observations for the given time series, in conformance with the ARIMA model. The predicted values are taken to estimate the forecasting upper and lower thresholds. Whenever the resultant entropy falls outside the prediction intervals, a Traffic Anomaly alert is reported to the auto-scaling engine of the corresponding Cloud platform (i.e., Openstack Heat).

7.3.3 Client-Side Component

On a separate server, several clients have been implemented as Python multi-threading scripts for HTTP traffic generation, which is sent to the web service hosted in the Openstack virtual machine instance. The generated number of traffic requests is a discrete variable that follows a random Poisson distribution, since their similarity with this distribution is widely assumed by the research community [BTI+03]. It is modeled according to the traffic load requirements for each evaluation scenario. Every client is represented by a process thread, which models multiple parallel clients handling their own sets of requests independently from others. When normal network conditions are modeled, all the clients send an HTTP GET request to the lower CPU-consuming requests (endpoints 1–3) described in Table 7.1. When an attacker is modeled, it only calls the most complex endpoint (4), which has higher CPU demands at server-side. Note that GET requests can also accept the client ID as a parameter. It facilitates the implementation of different client connections originated in the same computer since all the thread-based clients share the same source IP address, but are differentiated by client ID.

7.3.4 Test Scenarios

Five main scenarios have been showcased to validate the proposal. All of them compare the entropy levels of CPU processing times under normal traffic conditions against the entropy measured when an EDoS attack is launched. Those attacks target to produce CPU overhead. Therefore, the attack decrements the server capacities to handle more connections, and it forces the decision to scale up the current virtual machine instance when the CPU usage is above a pre-defined CPU limit in the Cloud-platform auto-scaling engine.

The set of network traffic conditions described in Table 7.2 are assumed throughout the experiments. There, clients (C) generate the total number of web requests (TR) at the expected rate (ERS). It is worth remarking that ERS corresponds to the expected number of occurrences (λ) of the Poisson distribution. Therefore, the generated web requests represent the sample of connections to be analyzed. The MTR observation number (5000) is the frontier that divides the TR into two groups of 5000 client requests each. The first one operates under the normal traffic conditions described in Table 7.2; whereas a percentage of the second group contains the malicious requests, letting the remaining connections to operate under the normal conditions. For instance, in the second group a 5% malicious requests rate indicates that 250 malicious requests and 4750 normal requests were observed. Table 7.3 defines the evaluation scenarios (E1 to E5) considered to deploy the EDoS attacks.

The experiments performed for each scenario started their execution with the normal web traffic conditions (first group of connections), with all the participant clients requesting the endpoints 1–3, as explained before. However, at the time specified by the MTR connection, the attack was launched. It compromised several normal clients (C), which sent malicious requests to the endpoint 4, thus increasing the CPU overhead. It is important to remark that the attackers connect to the server under the same ratio

Table 7.2: Normal traffic conditions for experiments.

Characteristic	**Value**
Web clients (C)	500
Expected requests per second (ERS)	60
Total web requests (TR)	10,000
Malicious Triggering Request (MTR)	5000

Table 7.3: Network attack conditions and scenarios.

Parameter	**E1**	**E2**	**E3**	**E4**	**E5**
Malicious Request Rate (MRR)	1%	5%	10%	15%	20%
Attacker Clients (AC)	5	25	50	75	100
Total number of malicious requests $(TR - MTR) \times MRR$	150	250	500	750	1000

(ERS) configured for normal clients, making them unnoticeable since their connection rate resembled legitimate traffic, but they targeted to exploit the highest time-consuming endpoint which was exposed as a service vulnerability. To validate the proposal, it has been considered a Cloud auto-scaling policy, configured to launch a new virtual machine instance when the CPU consumption ran above 40% in a one minute interval.

7.4 Results

The experiments were performed with the parametrization presented in Table 7.3, adapted to each evaluation scenario. The first monitored metric was the CPU time consumption caused to process web requests launched from clients. A summary of the CPU consumption of the server, measured on one-second intervals, is depicted in Figure 7.4. There, in all scenarios, half of the client connections exposed the same behavior until the attack was triggered (MTR). From that moment on, the CPU overhead was influenced by the traffic attack volume described in Table 7.3. Bearing in mind the defined auto-scaling policy, it is noted that the scenarios E3, E4 and E5 would have automatically launched a new virtual machine instance if the presence of the attack had been unnoticed. Hence demonstrating the consequences of the EDoS threats and bringing the attack detection strategy to play an essential role. On the other hand, besides the CPU estimation, the entropy of the per-client processing time was constantly measured by the Entropy Modeler on one-second intervals, as plotted in Figure 7.5. The graph shows that the overall behavior of the entropy was contrary to the behavior noticed in the CPU overhead with the higher entropy values before the MTR observation. The slumped entropy level was slightly noticeable on scenario E1 (Figure 7.5(a)), but became quite more perceptible on scenarios E2 to E5 (Figure 7.5(b) to 7.5(e)). Thereby, this pattern was directly influenced by the presence of the compromised devices, decreasing the entropy as long as more malicious requests were generated.

Only when the entropy was measured for the observed time, the Entropy Modeler

estimated the prediction thresholds to infer if the observed entropy was running outside the predicted intervals, thus leading to the decision of triggering an alert if the EDoS attack was detected.

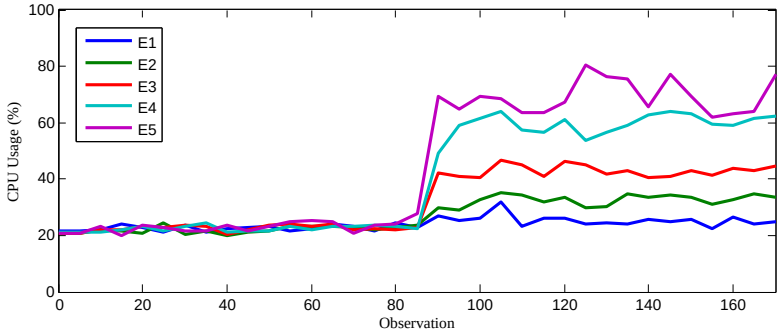
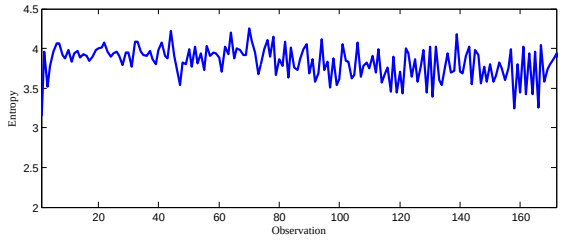


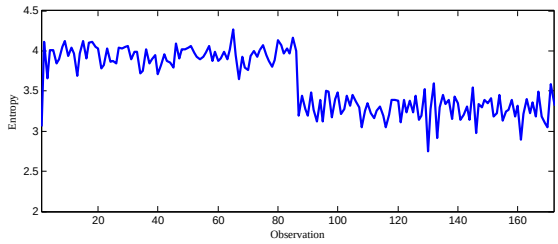
Figure 7.4: Average CPU consumption per scenario.

The precision observed at the Receiver Operating Characteristic (ROC) space is summarized in Figure 7.6. There five curves are illustrated, each one associated with one of the aforementioned evaluation scenarios (E1, E2, E3, E4, E5). Table 7.4 compiles several evaluation metrics (True Positive Rate (TPR), False Positive Rate (FPR) and Area Under Curve (AUC)) and the best calibrations (K) to reach the highest accuracy. Bearing in mind these results, it is possible to deduce that the proposed method has proven to be more effective when the attack is originated from a larger number of compromised nodes (e.g., E5 with 20% of the total number of connected clients). This is because a greater number of instances of the random variable X represent similar probabilities, which leads to a more significant decrease in the $H(X)$ entropy, and therefore to display less concordance with the normal observations.

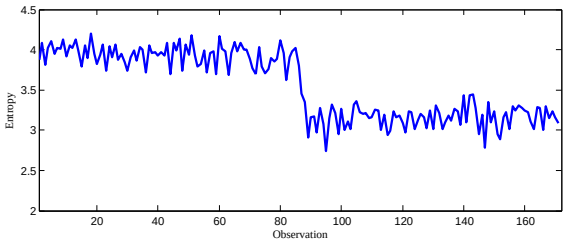
On the other hand, labeling errors have occurred mainly due to issuing false positives, in situations where fluctuations of $H(X)$ derived from changes in the behavior of legitimate clients acquire a similar relevance to those inferred by malicious activities. Note that the larger is the number of compromised nodes that take part of the attacks, the greater possibility of forcing auto-escalating reactions. Based on this fact it is possible to state that the proposed method improves its detection capabilities when facing more harmful threats. In addition, the existence of a K calibration parameter allows operators to easily configure the level of restriction in which the system operates: When greater discretion is required, K must adopt higher values. This considerably reduces the likelihood of issuing false alerts, hence facilitating to minimize the cost of the countermeasures to be applied. On the opposite case, when the monitoring environments require greater protection it is advisable to decrease K, hence improving the possibility of detecting threats, but potentially leading to deploy more unnecessary countermeasures.



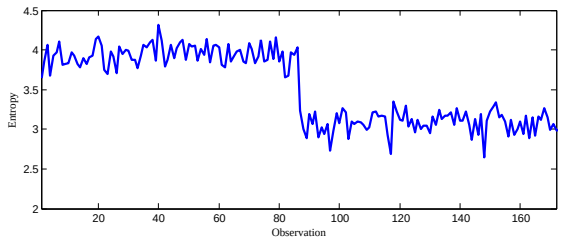
(a) Entropy evolution in E1



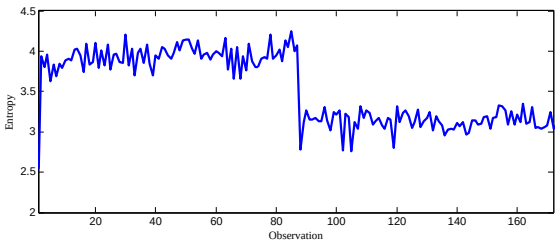
(b) Entropy evolution in E2



(c) Entropy evolution in E3



(d) Entropy evolution in E4



(e) Entropy evolution in E5

Figure 7.5: Entropy measurements per scenario.

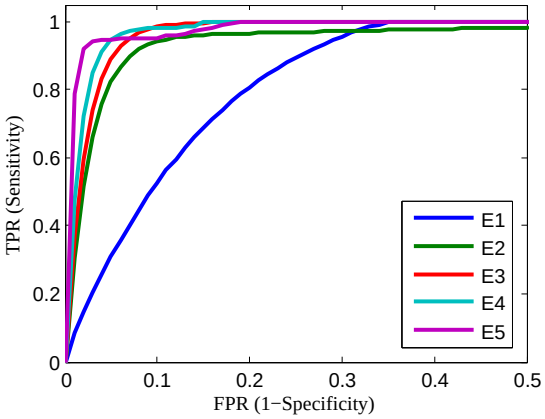


Figure 7.6: Results in ROC space.

Table 7.4: Summary of results in ROC space.

Scenario	AUC (Trapezoidal)	TPR	FPR	K
E1	0.8858	0.7480	0.17	0.160
E2	0.9637	0.9630	0.09	0.163
E3	0.9766	0.9680	0.08	0.160
E4	0.9794	0.9644	0.06	0.160
E5	0.9830	0.9431	0.03	0.167

7.5 Final Remarks

Throughout this chapter, a novel proposal for detecting EDoS attacks has been introduced. To this end, the state of the art has been reviewed in detail, which has brought several considerations towards the definition of a detection method suited for virtualized environments. A distinctive aspect of the performed research on this subject, compared to other proposals, is the study of behavioral entropy-based patterns analyzed on the deployed virtual instances. The experimental testbed implemented a client-server architecture executed on different network scenarios. On the web server, the monitored per-client CPU times have been evaluated by analyzing the entropy levels, which have exposed a decrement when malicious requests originated by the compromised nodes have been processed at server-side. In such scenarios, entropy has behaved indirectly proportional to the consumed CPU. In addition, the detection method has also demonstrated its effectiveness when predicting the entropy thresholds to be compared against the real measured entropy. Thereby, this approach has proven high accuracy by quantifying the area under the ROC curve, hence demonstrating its applicability on network environments where virtualization plays a major role.

Chapter 8

Detection of EDoS Threats in Self-Organizing Networks

In this chapter, the problem of the economic denial of sustainability in Self-Organizing Networks is discussed. To this end, the characteristics of these threats are identified and formalized. Thorough the performed research two novel threats were defined: workload-based EDoS (W-EDoS) and Instantiation-based EDoS (I-EDoS). W-EDoS is characterized by executing expensive requests in terms of computational resources at the victim system, hence exhausting its workload and forcing operators to contract additional resources. On the other hand, I-EDoS occurs when the cloud management software deploys more instances of virtual network functions than needed as a response to requests that resemble legitimate, but are malicious, thus increasing the cost of the hired resources. To contribute towards their mitigation, a security architecture is proposed. It implements strategies that rely on predicting the behavior of the protected system, constructing adaptive thresholds, and clustering instances based on productivity. An extensive experimentation has been successfully conducted, which includes case studies and the assessment of the accuracy under different scenarios.

For cooperating with the research community towards the detection and mitigation of EDoS, the following main contributions are presented:

- A comprehensive formalization of the distinction between EDoS threats and similar network incidents, among them normal activities, flash crowds and denial of service.
- The definition of a pair of emerging new generation threats: Workload-based EDoS and Instantiation-based EDoS.
- A multi-layered architecture for EDoS attack detection, which describes the management of the acquired information from its monitoring to the notification of possible threats.
- A novel entropy-based EDoS detection approach that allows to discover unexpected behaviors on local-level metrics related with the auto-scaling capabilities of the target system.

- An evaluation methodology adapted to the singularities of the EDoS threats and the assumptions driven by their original definition.
- Comprehensive experimental studies that validate the proposed detection method.

To facilitate the reader's comprehension, this chapter has been divided into nine parts. Section 8.1 introduces the CRoWN indicators that lay the basis for the formalization of this proposal. Section 8.2 describes the impact of EDoS attacks. Section 8.3 introduces the design principles and considerations of the proposed detection method. In Section 8.4 the self-organizing architecture is presented in detail. Sections 8.5 and 8.6 delve into the details of the detection approaches for dealing with W-EDoS and I-EDoS, respectively. Section 8.7 describes the experiments and evaluation methodology to assess the proposed method. Section 8.8 discusses the obtained results. Finally, Section 8.9 provides the conclusions of this chapter.

8.1 EDoS and CRoWN indicators

The principal disparities between EDoS attacks and the rest of similar network circumstances are easily understood by considering four essential elements involved at the information communication processes, under the prior assumption that the relationships between the sources of the requests and the provision that must to solve them act as a client-server model. These traits are the clients (C), requests (R), workload that entails their resolution (W) and the network functions necessary for their processing (NF), which are grouped in the set of characteristics $CRoWN : \{C, R, W, NF\}$. From the analytical point of view, and as deduced from the discussions reviewed in the bibliography, the quantitative study of their behavior has attracted the interest at most of the researchers. Therefore, the following CRoWN indicators are considered in the event definitions (see Table 8.1): with regard to C, the total number of monitored clients that made requests nC and its distribtuion over time $nC(t)$; with respect to R, the average number of requests per client nR and its distribution over time $nR(t)$; for W is considered the average effort towards process the requests nW (bandwidth, computational cost, memory, etc. depending the use case) and its distribution over time $nW(t)$; and finally, regarding NF the total number of network functions instantiated nNF, its distribution over time $nNF(t)$ and the productivity of each of them $P(X)_{i=0}^{nNF}$, $0 \leq i \leq nNF$. Note that the method for calculating the last indicator depends directly on the functionality of the instantiated network functions. For example, if they act as Network-based Intrusion Detection Systems (NIDS), a possible productivity indicator is the number of alerts that each of them report. On the other hand, if they deploy bandwidth optimization capabilities, productivity may be the improvement they achieve. It is also important to highlight that the aforementioned indicators can be further extended in order to define variations of the threats described throughout this research, introduce alternative network situations, or enhance the proposed method. But to address these issues in-depth is out of the scope of this paper, hence focusing on the features that most comprehensively recap

the problems to be faced. With this purpose, the CRoWN indicators associated to a monitoring task m are summarized in the following abstract data organization:

$$CRoWN(m) = [nC, nC(t), nR, nR(t), nW, nW(t), nNF, nNF(t), P(X)_{i=0}^{nNF}] \tag{8.1}$$

Table 8.1: Summary of CRoWN indicators

Trait	Indicator	Description
C	nC	Total number of clients.
	$nC(t)$	Distribution of clients over time.
R	nR	Average number of requests per client.
	$nR(t)$	Distribution of average number of requests per client over time.
W	nW	Total workload of processing requests.
	$nW(t)$	Distribution of workload of processing requests over time.
NF	nNF	Total number of instantiated network functions.
	$nNF(t)$	Distribution of number of instantiated network functions over time.
	$P(X)_{i=0}^{nNF}$	Productivity of network function i.

From them five categories of network situations related with EDoS at Self-Oganizing Networks are described: normal traffic [Cha09], flash crowds [BSMT14], flooding-based Denial of Service [ZWHL16], flooding-based Distributed Denial of Service [SGSC16], Workload-based EDoS and Instantiation-based EDoS (see Table 8.2). Two of them are legitimate (normal traffic and flash crowds) and the rest are malicious. Unlike Denial of Service, and Distributed Denial of Service, the EDoS threats satisfy the Network-based similarity condition. Because of this, the paper focuses on indicators related with the workload and productivity traits. Note that they are variants of the classical complexity-based DoS attacks [Afe16] that target the implementation of algorithms at software-level, but adapted to exploit Self-Organizing Network features. On this basis, let the network monitorizations A and B expressed bellow and the following situations:

$$CRoWN(A) = [nC_A, nC(t)_A, nR_A, nR(t)_A, nW_A, nW(t)_A, nNF_A, nNF(t)_A, P(X)_{i=0}^{nNF}{}_A] \tag{8.2}$$

$$CRoWN(B) = [nC_B, nC(t)_B, nR_B, nR(t)_B, nW_B, nW(t)_B, nNF_B, nNF(t)_B, P(X)_{i=0}^{nNF}{}_B] \tag{8.3}$$

Definition 8.1.1 *Flash Crowd characterization.*
Assuming that A is the monitorization of the habitual and legitimate network behavior, B is defined as flash crowd when $nC_A \ll nC_B$ and the rest of indicators display similar values. Hence let the $\mathcal{U}$ infinite set of possible network monitorizations according with CRoWN, and the Fc subset of flash crowd events, they are formalized as:

$$Fc_B \leftrightarrow \{A, B \in \mathcal{U} : nC_A \ll nC_B, \sim \textit{when other indicators}\} \tag{8.4}$$

Table 8.2: Network situations similar to EDoS attacks in SON

Nature	Family	Category	Description
Legitimate		Normal	Habitual traffic.
		Flash Crowd	Multitudinous agglomeration of legitimate requests.
Malicious	Flooding-based	DoS	Basic DoS attack from a simple source.
		DDoS	DoS attack from multiple sources.
	EDoS	Workload-based	Enforcing auto-scaling by costly requests.
		Instantiation-based	Enforcing the instantiation of network functions.

Definition 8.1.2 *Flooding-based DoS characterization.*
Assuming that A is the monitorization of the habitual and legitimate network behavior, B is defined as flooding-based Denial of Service (simple) or DoS when $nC_A \sim nC_B$, $nC_A(t) \sim nC_B(t)$, $nR_A \ll nR_B$ and $nR_A(t) \nsim nR_B(t)$, i.e. when the number of clients is similar to the normal behavior, but a significant increase in the average number of requests is observed. Let the $\mathcal{U}$ infinite set of possible network monitorizations according with CRoWN, and the DoS subset of Denial of Service events, they are formalized as follows:

$$DoS_B \leftrightarrow \{A, B \in \mathcal{U} : nC_A \sim nC_B, nC_A(t) \sim nC_B(t), nR_A \ll nR_B, nR_A(t) \nsim nR_B(t)\} \quad (8.5)$$

Definition 8.1.3 *Flooding-based DDoS characterization.*
Assuming that A is the monitorization of the habitual and legitimate network behavior, B is defined as flooding-based Distributed Denial of Service or DDoS when $nC_A \ll nC_B$, $nC_A(t) \nsim nC_B(t)$, $nR_A \ll nR_B$ and $nR_A(t) \nsim nR_B(t)$, i.e. when the number of clients and requests significantly increase. Let the $\mathcal{U}$ infinite set of possible network monitorizations according with CRoWN, and the DDoS subset of Distributed Denial of Service events, they are formalized as follows:

$$DDoS_B \leftrightarrow \{A, B \in \mathcal{U} : nC_A \ll nC_B, nC_A(t) \nsim nC_B(t), nR_A \ll nR_B, nR_A(t) \nsim nR_B(t)\} \quad (8.6)$$

Definition 8.1.4 *Network-based Similarity.*
As stated by Hoff [Hof08, Hof09], EDoS attacks pose great resemblance to the legitimate traffic. Henceforth, it is reasonable to assume that in this context, the number and distribution of clients and requests remain similar. So, let the A monitorization of the habitual and legitimate network behavior, B can only been categorized as Economic Denial of Sustainability attack if $nC_A \sim nC_B$, $nC_A(t) \sim nC_B(t)$, $nR_A \nsim nR_B$ and $nR_A(t) \sim nR_B(t)$. Hereinafter this relationship is referred as network-based similarity (abbreviated as NB), so if it is satisfied, it is possible to state that A and B are network-based similar. It is expressed as follows:

$$NB(A, B) \leftrightarrow \{A, B \in \mathcal{U} : nC_A \sim nC_B, nC_A(t) \sim nC_B(t), nR_A \sim nR_B \text{ and } nR_A(t) \sim nR_B(t)\} \quad (8.7)$$

Note that the variations on W and NF traits reveal if B pose a threat, hence distinguishing

it from the legitimate observations. From them, EDoS attacks on Self-Organizing Networks are classified in Workload-based EDoS attempts and Instantiation-based EDoS attempts. They are defined below.

Definition 8.1.5 *Workload-based EDoS characterization.*
Assuming that A is the monitorization of the habitual and legitimate network behavior, B is defined as Workload-based EDoS attack (abbreviated as W-EDoS) when A and B are network-based similar, $nW_A \ll nW_B$ and $nW_A(t) \nsim nW_B(t)$; i.e. when they pose strong resemblance at network-level, but the average workload per request significantly increased in B. In this way, the attacker exploits the auto-scaling capabilities of the protected system. Hence let the $\mathcal{U}$ infinite set of possible network monitorizations according with CRoWN, and the W-EDoS subset of Workload-based EDoS threats, the second is formalized as follows:

$$W\text{-}EDoS \leftrightarrow \{A, B \in \mathcal{U} : NB(A, B), nW_A \ll nW_B, nW_A(t) \nsim nW_B(t)\} \tag{8.8}$$

Definition 8.1.6 *Instantiation-based EDoS characterization.*
Assuming that A is the monitorization of the habitual and legitimate network behavior, B is defined as Instantiation-based EDoS attack (abbreviated as I-EDoS) when A and B are network-based similar, $nNF_A \ll nNF_B$, $nNF_A(t) \nsim nNF_B(t)$ and $P_A(t) \gg P_B(t)$; i.e. when they pose strong resemblance at network-level, there is a significant increase of network function instances in B, but their overall productivity decrease. Hence, the attacker takes advantage of the actuation capabilities of the Self-Organizing Networks by deploying useless functionalities. Let the $\mathcal{U}$ infinite set of possible network monitorizations according with CRoWN, and the I-EDoS subset of Instantiation-based EDoS threats, they are formalized as follows:

$$I\text{-}EDoS \leftrightarrow \{A, B \in \mathcal{U} : NB(A, B), nNF_A \ll nNF_B, nNF_A(t) \nsim nNF_B(t), P_A(t) \gg P_B(t)\} \tag{8.9}$$

8.2 EDoS Impact

The following subsections describe the distinctive aspects of EDoS attacks from the perspective of a cloud platform.

8.2.1 Impact of Workload-based EDoS

Following the formalization stated in Definition 8.1.5, a W-EDoS attack is characterized by the execution of costly operations on servers hosted in a cloud provider. Server-side operations are produced by client requests that resemble legitimate traffic in terms of network indicators considering not only the number of clients, requests, workload or deployed network functions, but also their distribution over time. The effect of W-EDoS attacks is the need to scale-up or scale-out the deployed cloud instances (typically VNFs), by adding additional computational resources when the existing ones prove insufficient to ensure the desired quality of service, in conformance with the configured scaling policies.

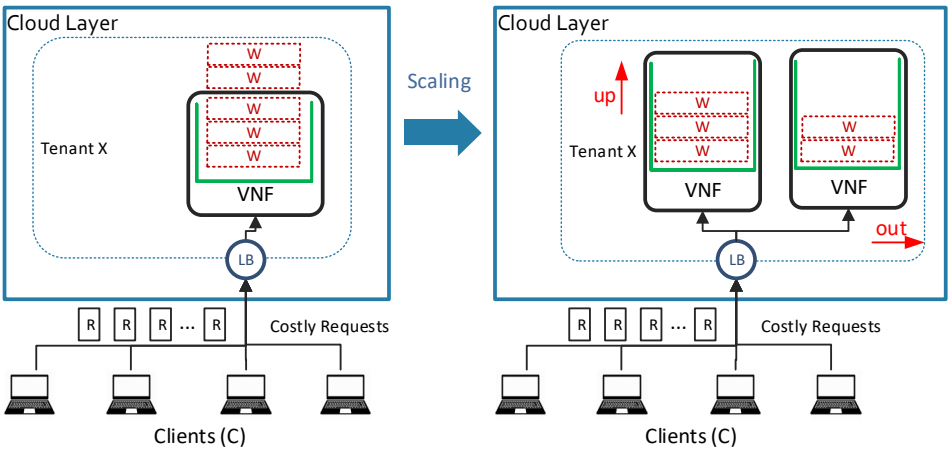


Figure 8.1: Auto-scaling triggered by a W-EDoS.

The major drawback of these auto-scaling enforcements is the associated monetary cost derived from the allocation of the new resources. Figure 8.1 illustrates the scaling process of a running an additional VNF enforced by a W-EDoS attack. In the depicted scenario, the VNF has insufficient resources to process the workload (W), leading to a scale-up or scale-out strategy which accommodates a proper task processing, balanced in several instances of the same VNF.

8.2.2 Impact of Instantiation-based EDoS

Assuming the ability of the attacker to exploit a vulnerability that triggers the automatic deployment of VNF instances in distributed locations of the infrastructure, an I-EDoS attack occurs when the cloud management software deploys more VNF instances than needed as a response to one or more requests ostensibly legitimate, thus increasing the cost of the hired cloud resources due to the rise on the number of instances, with a decrement on the average service productivity. This scenario fits with the principles described in Definition 8.1.6. Figure 8.2 illustrates an I-EDoS attack where the cloud platform itself, or a specific network function, is assumed to have an auto-scaling policy vulnerability exposed to the attackers, and serves in consequence as an entry point of malicious requests. They generate the automatic allocation of several virtual instances in different network locations, but rather than being efficient they show scarce productivity levels. Those lazy VNF instances are thereby unnecessary by the network operator since they generate useless expenses without adding real value to the provided service.

8.3 Design Principles

The method described in this section addresses the problem of distinguishing legitimate monitored situations from those related to attempts of Economic Denial of Sustainability in Self-Organizing Networks. In order to limit the tasks involved on both design and development stages, the following capabilities and restrictions have been assumed as design principles.

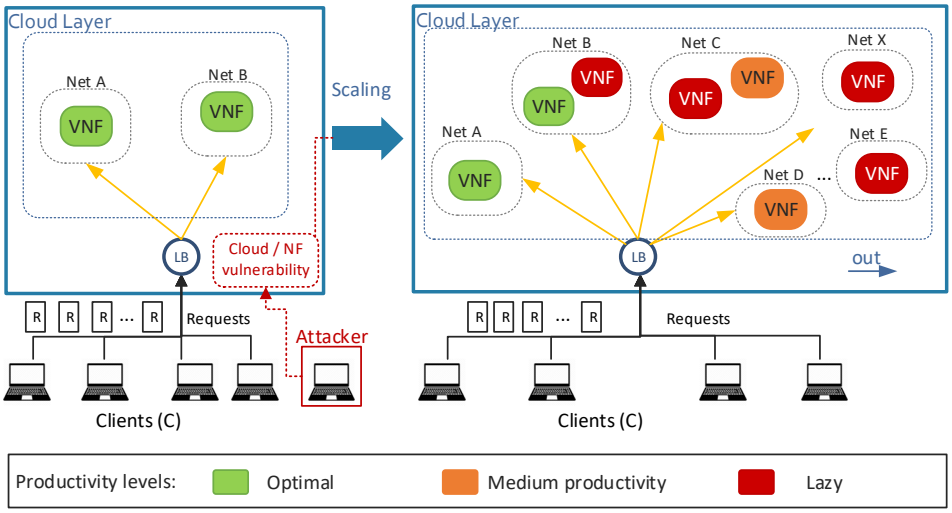


Figure 8.2: Auto-scaling triggered by a I-EDoS.

- The description of the network situations summarized in Table 8.2 and their descriptions assuming CRoWN indicators are considered. On this basis, the proposed method must be capable of successfully identifying W-EDoS and I-EDoS attacks, as well as distinguishing them from legitimate activities (normal traffic and flash crowds).

- The detection of conventional flooding-based DoS and DDoS attacks is out of the scope of this publication. Currently there is a large bibliography that facilitates its recognition [ZJT13, BBK15], hence principally aiming on contributing at the EDoS threat mitigation.

- The introduced approach considers a non-stationary monitoring environment [DRAP15]. This is feasible as justified in publications like [BTI+03], which are widely supported by an important part of the research community. Note that during its development, the importance of the adaptation to changes in the monitored data has been taken into account, which plays an essential role towards avoiding situations related with the *concept drift* [EP11]. However, these capabilities make this proposal difficult to understand, in this way diverting the attention from the security problem to be solved (i.e. its main goal).

- For reasons like those discussed above, adversarial methods based on imitation or identity theft [ÖB15, ADAH14] with the purpose of evade the proposed detection strategy are not considered.

- Self-Organizing Networks are complex monitoring scenarios where a large number of sensors collect information about the state of the network in real time. This information must be aggregated into observations that can be handled by high-level analytic tools. Although in the experimentation the impact of the granularity with which data is extracted is briefly reviewed, the introduction of methods for its calibration is postponed to future investigations.

- The correlation and management of the alerts discovered [SMFDV13, MAJ13] is out of the scope of this publication. However, the acquired knowledge must be notified with all the metadata required for their post-processing.

- The proposal aims to illustrate the detection method by focusing on the analytical approach. The impact of security measures implemented on the developed software are thus not considered when assessing the EDoS detection accuracy. Hence,it is postponed for future work.

8.4 Architecture

This section describes the EDoS detection architecture, which is illustrated in Figure 8.3. It has been designed in conformance with the ETSI-NFV [ETS13] and 5G architectures [5G-16b], where the decoupling of data and management planes makes it possible to distinguish the division between functional layers. The Physical sublayer is typically composed by Commercial-Off-The-Shelf (COTS) hardware on which the Virtualization Sublayer acts as a core component for the creation of virtual resources such as hosts, network links, storage elements and so on. On top of that, a Cloud Layer manages the automatic instantiation of Virtual Network Functions (VNFs) by interfacing with the Virtualization Sublayer, thus allowing the dynamic provision of resources needed to fulfill the agreed service levels. The deployed cloud environment interconnects VNFs through the underlying virtual network, composing in turn a forwarding graph where VNFs can also be chained with Physical Network Functions (PNFs), thereby creating one or more Network Services (NS) offered to the users. Cloud deployments are also isolated between customers to allow the coexistence of different network operators sharing the same physical resources, which is known as multi-tenancy support. The Cloud Layer gathers also an important number of metrics (e.g. usage or performance) measured on the instantiated resources allocated into the existing tenants. Besides that, sensors represent key components in the process of monitoring SON networks since they target to monitor customized network metrics, mainly at application-level. In this way, primary information collected by sensors and the cloud platform itself lies the basis to perform complex analysis at VNF level in the SON Autonomic Layer, whose subcomponents are described in the forthcoming sections.

8.4.1 Data Collection

In this module, the monitorization of the protected environment is carried out to extract raw metrics regarding the operational conditions of the virtual cloud. Notwithstanding the vast number of counters provided by most cloud platforms (i.e. Openstack [Ope], AWS [AmW], Azure [WMs]), they might be insufficient when more complex analysis tasks are required. Therefore, a SON-driven approach can facilitate the extraction of more specific information gathered by sensors, hardly feasible at cloud-management-level, to enhance the analysis not only for EDoS attacks, but also for several other scenarios. The Data Collection module comprises the following submodules.

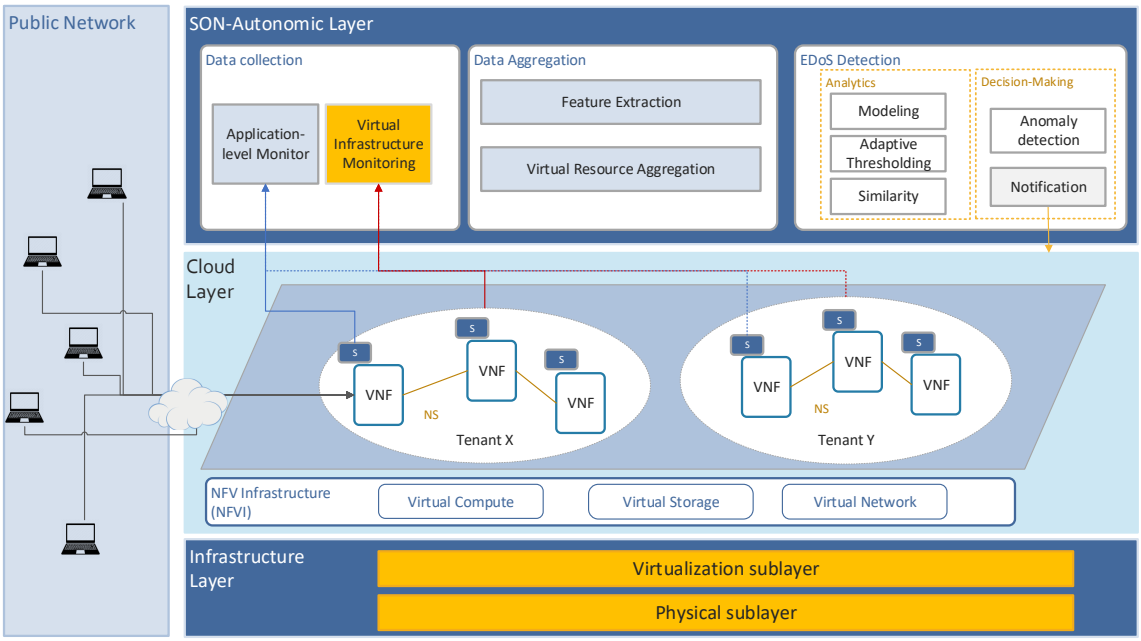


Figure 8.3: Self-organized EDoS Detection Architecture

- *Application-level monitoring.* VNFs are monitored by sensors at operating system or application-level to extract specific metrics targeted to enable further analytical processes in the SON Autonomic layer. For instance; service response time, memory consumption per process, number of active connections, among other counters.
- *Virtual Infrastructure Monitoring.* Built-in cloud-monitoring services, such as Openstack Telemetry (Ceilometer) [WOS], produce several metrics related to the virtual infrastructure (i.e. CPU, memory or network usage). Some of them are obtained periodically while others are generated when certain events are triggered (i.e. creation of a new virtual network).

8.4.2 Data Aggregation

The monitorization of cloud deployments generate huge volumes of data, including counters, events, alerts, among others. To facilitate their analysis, data aggregation operations are applied to produce a single metric that summarizes a collection of sampled observations, thus reducing the existing data volume. Different data granularity levels should be also possible to configure, in accordance with the intended analysis. The Data Aggregation module comprises the following submodules:

- *Feature extraction.* A batch of raw observations generated by the sensors is grouped on regular time intervals, when the configured aggregation operation is applied on those values. This leads to the generation of a time-series of aggregated metrics; for example, the entropy degree in the W-EDoS analysis.
- *Virtual Resource Aggregation.* The virtual infrastructure metrics collected are also summarized with aggregation operations. Most of the polled metrics might be easily

expressed as time series since they can be queried by the associated timestamp registered on the monitoring database. For instance, average CPU consumption in Ceilometer.

8.4.3 EDoS detection

This module performs the essential analysis tasks for the detection of EDoS attacks, and it is composed by the following submodules:

- *Modeling.* With the time series of aggregated metrics, a forecasting model is created to estimate the next m values of the time series by the application of well-known prediction algorithms.
- *Adaptive Thresholding.* When compared with real observations, a prediction interval (PI) is generated based on the forecasting error measured at each observation. Unexpected behaviors are inferred when the value of an analyzed metric runs outside the boundaries of the upper or lower prediction thresholds.
- *Clustering.* Clustering methods are intended to separate groups of virtual instances by analyzing similarities on their productivity indicators. It leads to the distinction of a suspicious group of instances whose creation does not resemble a legitimate pattern.
- *Anomaly detection.* Discordant behaviors exposed by the forecasting and clustering processes are analyzed by a rule-based system to infer conclusions about possible anomalies. When rule matches are found, they are labeled either as W-EDoS or I-EDoS attacks depending on the analyzed scenario.
- *Notification.* The inference of anomalies is reported to the cloud-auto-scaling engine. It aims to prevent the creation of VNF instances that are targeted to overspend the use of virtual resources, and the operational cost in consequence.

8.5 Workload-based EDoS recognition

The following describes in detail the metrics required for W-EDoS detection, and how they are analyzed for identifying unexpected behaviors and deciding if they should be tagged as potential threats.

8.5.1 W-EDOS Metrics

As stated in Section 8.3, the W-EDoS threat satisfies the *network-based similarity*, but unleash a significant variation in terms of workload (at both W and $W(t)$); in particular, concerning those metrics directly related with the auto-scaling capabilities of the hosts that support the task involved in solving requests [BBBS17]. Because of this, the proposed approach considers as W-EDoS detection metrics, the CPU consumption (X_{cpu}) and response-time at application-level (X_{app}). The first one measures the CPU usage at

the operating system-level, and the second focuses specifically on each service request consumption. Although a direct relationship between X_{cpu} and X_{app} is expected, this is not always the case. The nature of the requests can lead to a different workload in each of these levels, in this way being able to force self-scaling [SGSC16]. Because the proposed method lies on the detection of unexpected behaviors by observing the aforementioned metrics, the first step is to discover the response-time variations. This is carried out by studying the disorder degree on the application-level metrics reported once the different tasks are fulfilled. As is commonly approached in the bibliography, this is addressed on the basis of [JSTD16, ÖB15], where the gathered information is correlated in terms of entropy. As indicated by Bhuyan et al. [BBK15], among the various proposals, that defined by Rènyi provides a general-purpose solution, so it was implemented for W-EDoS detection. Hence $H_\alpha(X_{cpu})$ defines the Rènyi entropy and α is the Rènyi entropy order, $\alpha \geq 0$, $\alpha \neq 1$. The result is expressed as follows:

$$H_\alpha(X_{app}) = \frac{1}{1-\alpha} \log \sum_{i=1}^{n} P_i^\alpha \tag{8.10}$$

Note that given that Rènyi entropies are in range $[0, \log n]$, they are normalized as $H_\alpha(X_{app})/\log n$. The implemented method considers the normalized version.

The study of the impact of X_{cpu} and X_{app} can be approached as a classical Statistical Process Control (SPC), where the metrics monitored over time are modeled as time series. In this context they are univariate, since it was previously assumed that it is not possible to state that $X_{cpu} \perp\!\!\!\perp X_{app}$. Hence, the Rènyi entropies calculated through the performed application-level observations relate with each other as $H_\alpha(X_{app})_{t=0}^{n}$ sequences, where:

$$H_\alpha(x_{app})_{t=0}, H_\alpha(x_{app})_{t=1}, \cdots, H_\alpha(x_{app})_{t=n} \tag{8.11}$$

And the CPU consumption is represented as the following time series:

$$(x_{cpu})_{t=0}, (x_{cpu})_{t=1}, \cdots, (x_{cpu})_{t=n} \tag{8.12}$$

Note that for simplicity and taking into account that the rest of the analytics for W-EDoS are similar for both time series, X_{cpu} and X_{app} are refereed as the random variable X, hence not assuming distinctions.

8.5.2 Workload-based unexpected behaviors

The W-EDoS detection method lies on deciding if the estimation $\hat{X}_{t=m}$ at m horizon, $H > n$, significantly differs from $X_{t=m}$. Hence it is required to forecast the time series of the current $X_{t=0}^{n}$ observations until a predefined horizon is reached, then being able to complete the comparison (see Figure 8.4). Among the several approaches to this problem [Mak00], the W-EDoS detection implements the Double Exponential Smoothing (DES) [GJD80]. This decision has taken into account three conditions: firstly, non-seasonality is assumed as design principle, hence leaving aside more complex models like Holt-Winters [Gro73], which are capable of adapting to these variations. On the other hand, the system

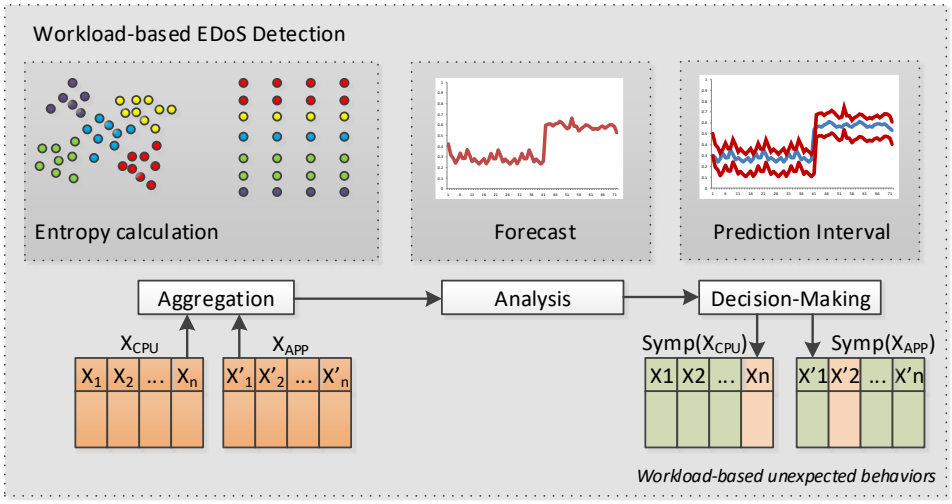


Figure 8.4: Workload-based EDoS detection process.

must be efficient and requires short time series for warmup and modeling, which leaves aside autoregressive approaches such as ARIMA [HT82].

By definition, the Simple Exponential Smoothing (SES) algorithm may not operate effectively when there is a trend in the analyzed time series [Bro57, Hol04]. In addition to the smoothing constant α inherited from the SES model (in order to disambiguate this value with respect to the Rènyi entropy degree, henceforth refereed as A, DES considers the constant γ related with the trend smoothing factor, so $0 < A), \gamma < 1$. It is calculated by the following recursive equations:

$$S_t = Ax_t + (1 - A)(S_{t-1} + b_{t-1}) \tag{8.13}$$

$$b_t = \gamma(S_t - S_{t-1}) + (1 - \gamma)b_{t-1} \tag{8.14}$$

where S_t is the level of the time series at t, and b_t is the trend. Note that when $A = 0$, DES becomes a naïve forecasting approach, and if A, $\gamma = 1$ it acts as SES. As is frequent in the bibliography, the base cases are the initializations $S_1 = \gamma_1$ and b_1, so:

$$b_1 = \frac{x_{t=n} - x_{t=1}}{n - 1} \tag{8.15}$$

and the forecasts are calculated as follows:

$$\hat{X}_{t=n+1} = S_t - b_t \tag{8.16}$$

$$\hat{X}_{t=n+m} = S_t - mb_t \tag{8.17}$$

The prediction intervals define the adaptive thresholds considering [MWH97], and as suggested in [HKOS05]. They are expressed based on the ϵ_t prediction error based on the Mahalanobis distance at t, in particular when $t = m$.

$$\epsilon_t = \sqrt{(x_m - \hat{x}_m)^2} \tag{8.18}$$

In this way the following adaptive threshold is built:

$$PI = x_{t=n} \pm \eta\sqrt{\sigma^2(\epsilon_t)} \tag{8.19}$$

where the parameter η, $\eta \geq 0$ calibrates the restrictiveness of the prediction interval and m is the forecast horizon. In the context of the W-EDoS detector, the significance of the forecasting errors can be assessed according to the Definition 8.5.1.

Definition 8.5.1 *Workload-based unexpected behaviors.*
Let $X_{t=0}^{n}$ and its forecast $\hat{X}_{t=n+m}$ at horizon m, an observation $X_{t=n+m}$ is workload-based unexpected when $\epsilon_t \notin PI$ is satisfied, i.e. when $\hat{x}_{t=n+m}$ and $x_{t=n+m}$ differ significantly. The sequences of unexpected observations are referred as workload-based unexpected behaviors.

The persistence of observations tagged as *workload-based unexpected behaviors* establishes the duration of the possible threats. An example of this situation is illustrated in Figure 8.5, where $X_{t=0}^{n}$ is analyzed and the prediction intervals are built. At the range of observations ($x_{t=41}$, $x_{t=44}$) they are overtaken, in this way displaying four *workload-based unexpected* observations ($x_{t=41}$, $x_{t=42}$, $x_{t=43}$ and $x_{t=44}$) and the *workload-based unexpected behavior* $X_{t=41}^{44}$.

Alerts related with W-EDoS are reported when at the same observation t, both detection metrics (X_{cpu} and X_{app}) are *workload-based unexpected*. Note that given the not $X_{cpu} \perp\!\!\!\perp X_{app}$ relationship, each of them may deduce a suspicious outlier, being $Symp(X_{cpu})$ related with abusive CPU workloads, and $Sympt(X_{app})$ being related with anomalous time-responses at application-level. The decision of pooling both symptoms lies on filtering the rebound effect of the adaptive thresholding method, where when a representative change occurs in the time series, the threshold takes a while to re-calibrate, which may lead to the emission of false positives as replicas. This is clearly illustrated in Figure 8.6, where after discovering an outlier ($t = 47$), the prediction interval is erroneously calculated at the next monitorizations ($t = 48$ to $t = 65$). On the other hand, a $Symp(X_{cpu})$ may be triggered by anomalous decrements on the workload of the monitoring systems, which leads to an effect contrary to that caused by the W-EDoS. This is illustrated in Figure 8.7(a) where $H_\alpha(X_{app})$ displays a discordant behavior because of changes at the disorder degree on the application-level; in particular the entropy decreased because most of the host supporting certain service are removed driven by a decrease of the demanded workload Figure 8.7(b).

8.6 Instantiation-based EDoS recognition

The subsections below detail the metrics studied by the I-EDoS detection components, how they are analyzed and the decision-making task.

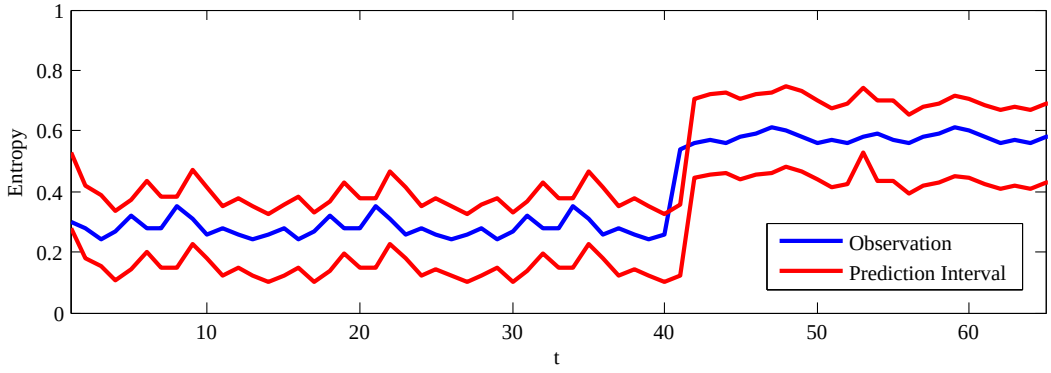


Figure 8.5: Example of workload-based unexpected behavior.

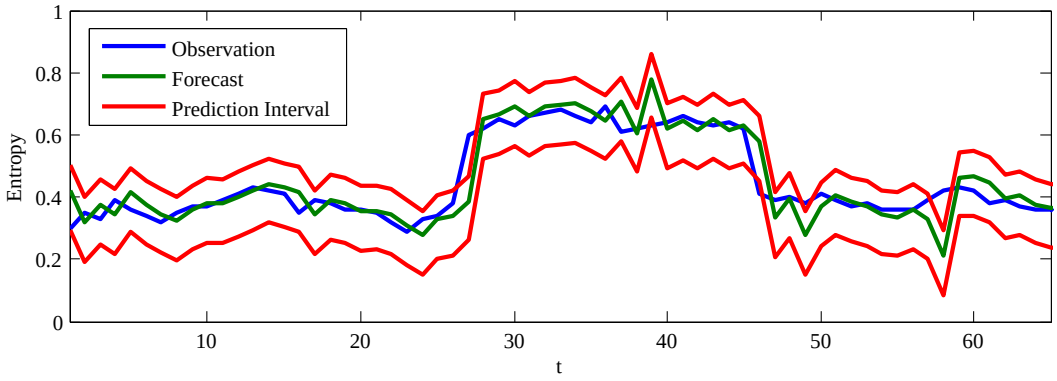
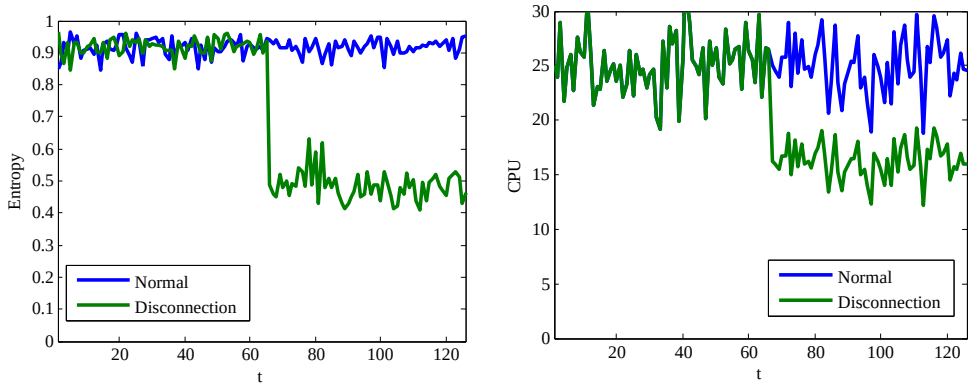


Figure 8.6: Example of rebound effect.



(a) Entropy decrease (b) Workload decrease

Figure 8.7: Example of discordant entropy not related with W-EDoS

8.6.1 I-EDOS Metrics

As indicated in Definition 8.1.4, I-EDoS situations are *network-based similar* with the normal and legitimate traffic, but entail outliers in terms of nNF, $nNF(t)$ and $P(X)_{i=0}^{nNF}$. It was also stated that the classical instantiation-based attacks are characterized by the emergence of a significant number of network functions behaving in low-productivity. Therefore, before labeling a situation as possible threat, there must be observed a direct

relationship between the new network functions and their low productivity. This leads to consider a pair of I-EDoS detection metrics: the number of network functions instantiated per observation Y, and their productivity Z; where Z is the set $Z = \{z_1 \cdots z_Y, Y \geq 0\}$ that defines the productivity of the different network functions $\{z_1 \cdots z_Y\}$ at certain t observation. In analogy to the W-EDoS, they are monitored over time and collected in the following times series:

$$Y_{t=0}, Y_{t=1}, \cdots, Y_{t=n}; (Y_{t=0}^{n}) \tag{8.20}$$

$$Z_{t=0}, Z_{t=1}, \cdots, Z_{t=n}; (Z_{t=0}^{n}) \tag{8.21}$$

On this basis, an observation at t, $0 \leq t \leq n$, is potentially malicious when Y_t has grown discordantly and $Z_t = \{z_1, \cdots, z_{Y(t)}\}$ displays a group of instances that clearly behave at low productivity; in particular, the low-productive functions must be those that triggered the increase of Y_t, in this way satisfying $Y \perp\!\!\!\perp Z$. These situations are defined below.

8.6.2 Novelty detection

The anomalous growth of the number of instances are studied similarly to the *workload-based unexpected behaviors*, i.e. by observing if their projection at m differ significantly from the observations at m. With this purpose, and in the grounds stablished at Section 8.5.2, the Double Exponential Smoothing (DES) [GJD80] is implemented, and adaptive thresholds are built according with [HKOS05]. Consequently, a novelty in terms of number of instances Y at t is considered as potential trait of I-EDoS when a significant growth is observed in Y_t, which is formalized according to Definition 8.6.1.

Definition 8.6.1 *Significant growth.*

Let $Y_{t=0}^{n}$ and its forecast $\hat{Y}_{t=n+m}$ at horizon m, an observation $Y_{t=n+m}$ implies a significant growth when $\epsilon_t \notin PI$ is satisfied, i.e. when $Y_{t=n+m}$ and $\hat{Y}_{t=n+m}$ differ significantly, and $Y_{t=n} < Y_{t=m}$.

8.6.3 Identification of suspicious network function instances

Once a self-organizing action at t has resulted in the instantiation of $Z_t = \{z_1, \cdots, z_{Y_t}\}$ new network capabilities, it is possible to detect if they are involved in an I-EDoS attack. According to the threat definitions discussed in Section 8.1, the typical behavior of the malicious instances is low-productivity. But low-productivity may be the pattern observed in most of the network functions deployed; therefore, it is assumed that the malicious instances should register significantly low-productivity with respect to the rest of the instances that operate on the monitoring environment at t. How to detect this situation is not a simple question. But despite detecting this situation is not trivial in many of the network scenarios, it can be addressed from different perspectives. Among them, the performed research has focused on those based on clustering, thus leaving other possible approaches for future investigation. At the experimentation, a density-based approach

was implemented; in particular, the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [EKS+96].

DBSCAN considers groups of observations based on the density of the K-nearest neighbors. This process distinguishes three kinds of o objects in the space: core objects, density-reachable and outliers. Let the object p, it is a core object if the ϵ-neighborhood of the object contains at least $minPts$ objects; where the ϵ-neighborhood of p is the space within a radius *epsilon* centered at p. Because of this they are considered the pillars of dense regions. The objects within ϵ are considered directly density-reachable by p. On the other hand, a q object is reachable by p if there is a chain of objects $p_1, \cdots, p_n$, where $p_1 = q$, $p_n = q$; and each p_{i+1} is directly density-reachable with respect to ϵ and $minPts$, $0 \leq i \leq n$, $p_i \in D$. Each cluster has at least a core point, being the rest of the points their periphery. Consequently, the objects within a cluster are interconnected, and if p is density-reachable by other object q, the second also is part of the same group. Observations non-reachable by objects in clusters are considered outliers [Cha09]. DBSCAN has several advantages, which played an essential role in its selection as support for deploying an effective defense against I-EDoS, among them high efficiency, not requirement of previous estimations of the number of clusters to be defined, and noise tolerance. However, among other drawbacks it is worth mentioning its efficiency, which highly depends on the distances and similarity measures adopted. With the purpose of discover I-EDoS threats, DBSCAN is performed on each $Z_t = \{z_1, \cdots, z_{Y_t}\}$ set, where each productivity z_i is an object p_i, so $Y_t = n$. The following ϵ distance based on the Mahalanobis divergence was implemented:

$$\epsilon = \sqrt{(z_i - z_j)^2} \tag{8.22}$$

where $0 \leq i, j \leq n$ and $i \neq j$. The resultant set of K clusters is $C_t = \{c_1, \cdots, c_k\}$, $K \geq 0$ were $c_i = \{z_r, \cdots, z_s\}$, $0 \leq r, j \leq n$, i.e. $z_r, \cdots, z_s$ are network functions with similar productivities. The members of C_t can be increasingly ordered according to the average productivity of their members, being each $\bar{c_i}$ the mean of $z_r, \cdots, z_s$. The arranged list is expressed as $s(C_t) = [c_1, \cdots, c_K]$, $\forall c_i : c_j$ $0 \leq i \leq j \leq K$ is satisfied $\bar{c_i} \leq \bar{c_j}$. The clusters relevant for the I-EDoS detection are tagged as *lazy groups* and their members are termed *lazy instances*, which are defined in Definition 8.6.2. An example of them is illustrated in Figure 8.8, where it is possible to observe a subset of instances which productivity is significantly low.

Definition 8.6.2 *Lazy instances and groups.*

Let $Z^n_{t=0}$, for each $Z_t = \{z_1, \cdots, z_{Y_t}\}$ clustered into $C_t = \{c_1, \cdots, c_k\}$, $K \geq 0$, and sorted resulting $s(C_t) = [c_1, \cdots, c_K]$; c_1 is defined as the lazy group of network function instances, hence $\forall c_j : c_j \in s(C_t), 1 < j \leq K$ and the inequation $\bar{c_1} \leq \bar{c_j}$ is satisfied. Each member of c_1 is considered a lazy instance, and will play a relevant role when deciding if the protected environment is suffering an I-EDoS attack.

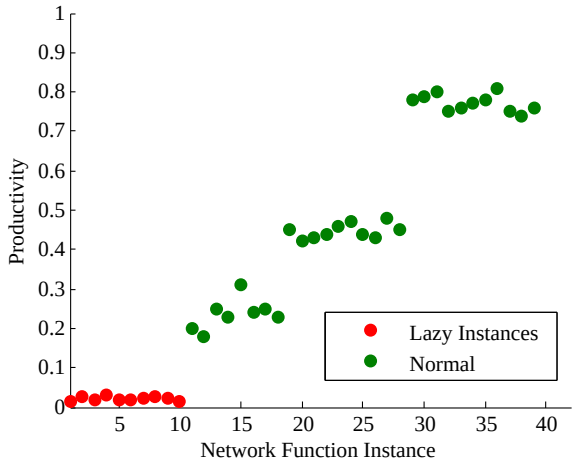


Figure 8.8: Example of lazy group of instances.

8.6.4 Instantiation-based unexpected behaviors

Ideally, it is expected that when a I-EDoS attack occurs a discordant increase in the number of instantiated network functions is observed. Consequently, and assuming that the attack is launch at the observation registered at t, a *significant growth* (see Definition 8.6.1) in the number of instances must be detected on Y_t. On the other hand, the network functions triggered by the attacker must behave with low productivity. Thus, most of them must be part of the lazy group of $Z_t = \{z_1, \cdots, z_{Y_t}\}$, in this way being lazy instances. When both conditions are satisfied, a potential I-EDoS threat was recognized at t.

But it is important to bear in mind that in real circumstances, these observations may occur asynchronously. This is because each type of network function taken into account in Z_t may require a different adaptation period, which is usually referred as warm-up time. Therefore, they are not expected to be productive until the warm-up time expires. A clear example of this situation can be comprehensively illustrated with the following example: let a SON configured to deploy intelligent agents for load balancing purposes similar to those introduced in [CSJN05]. These actuators are triggered once the number of clients connected to certain service overtake a previously defined static threshold. The productivity metric considered for I-EDoS detection is the Key Performance Indicator (KPI) that aggregates the server throughput in terms of GB/s, i.e. the amount of data transferred to and from the load balancer. These intelligent actuators require a warm-up time to collect reference data and build prediction models. Now suppose that the intruder knows the threshold that triggers the generation of new agents, and that by registering malicious clients, it is able to cause an I-EDoS attack. In the observations that this occurs a *significant growth* of the number of instances is registered, but they are not able to report productivity, since they are initializing their internal network usage models. Because of this, the definition of the *instantiation-based unexpected* behaviors related with I-EDoS threats must take into account the creation date of the instances and from when they are expected to begin to be productive. This leads to Definition 8.6.3, which stablishes the condition that must be satisfied prior to report possible I-EDoS incidents (see Figure 8.9).

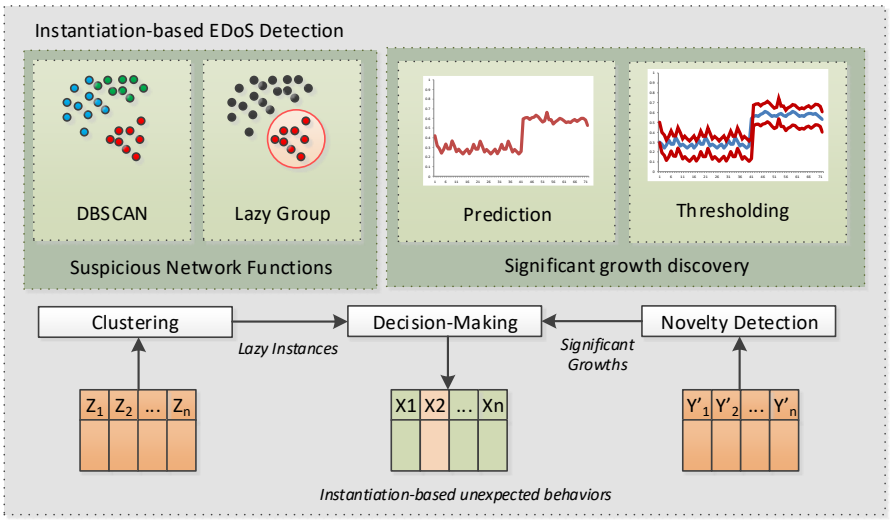


Figure 8.9: Instantiation-based EDoS detection process.

Definition 8.6.3 *Instantiation-based unexpected behaviors.*
Let the set of instances $Z_t = \{z_1, \cdots, z_{Y_t}\}$ *observed at* t*, clustered into* $C_t = \{c_1, \cdots, c_k\}$*,* $K \geq 0$*, and sorted resulting* $s(C_t) = [c_1, \cdots, c_K]$*. They are instantiation-based unexpected when there was registered a significant growth at the creation date of most of the members of their lazy group* c_1*. The set of instantiation-based unexpected observations over a monitorization process is referred as instantiation-based unexpected behavior.*

Therefore, when some observation in terms of the detection metrics (Y, Z) is tagged as *instantiation-based unexpected*, the proposed systems reports a possible I-EDoS situation $Symp(Y, Z)$. The persistence of events tagged as *instantiation-based unexpected behaviors* establishes the duration of the possible threat.

8.7 Experiments

This section describes the SON environment where the Cloud layer is complemented with the SON Autonomic components, thus extending the analytical capabilities of Cloud-platforms with more advanced features targeted to mitigate EDoS attacks. In addition, the performed tests carried out to detect W-EDoS and I-EDoS attacks are described in detail.

8.7.1 TestBed

The experimentation testbed was deployed to match the architectural layers defined in Section 8.4. This SON-oriented scheme is illustrated in Figure 8.10. The Cloud Layer has been implemented with Openstack [Ope], a widely used opensource platform to manage the lifecycle of small and large-scale cloud environments. Openstack has been deployed on two nodes: controller and compute (Nova). Each of them runs core Openstack functionalities in conformance with the NFV infrastructure layer. Moreover, additional

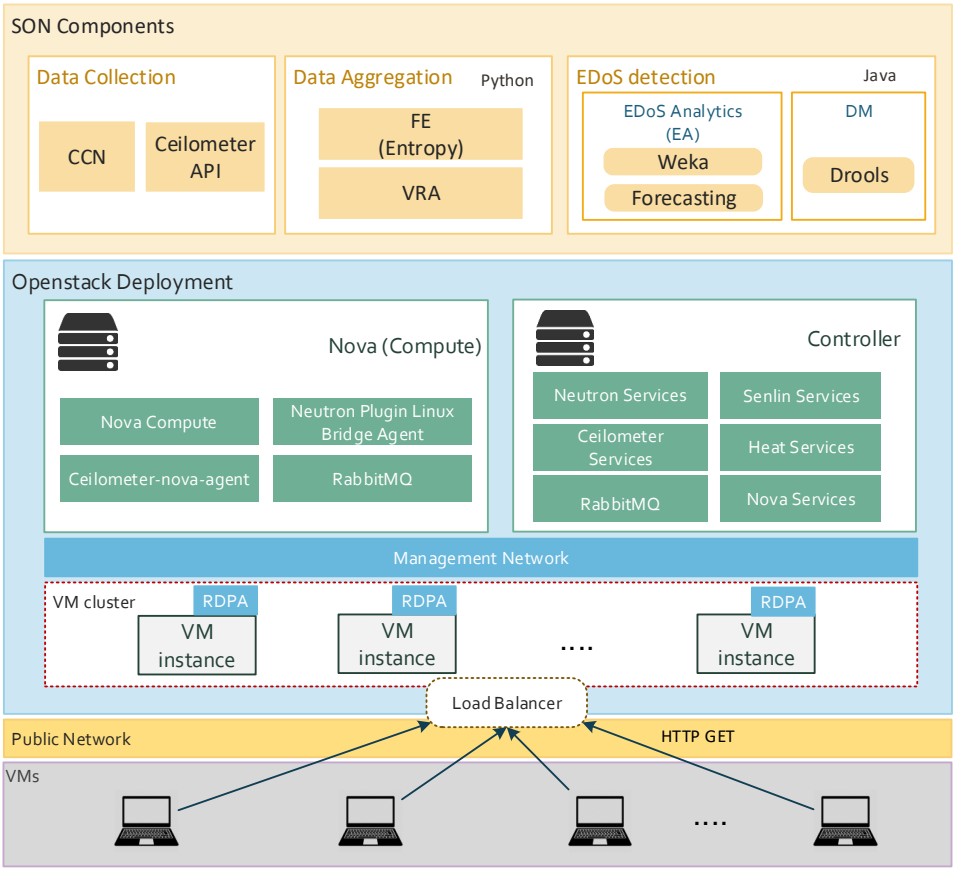


Figure 8.10: Testbed

features oriented to include auto-scaling mechanisms have been included to showcase EDoS attacks. The controller node hosts also the networking service (Neutron), Nova essential features, and RabbitMQ [RbM], the message broker software that allows the inter-process communication between the Openstack services. The compute node hosts also the Clustering (Senlin) [OSL] and Orchestration (Heat) services required to configure scaling policies. In addition, the compute node hosts the Telemetry service, intended to measure cloud-platform statistics with monitoring, scaling and billing purposes. All the Openstack services are interconnected through a private management network, in this way making possible to manage the on-demand instantiation of virtual resources to conduct the experimental EDoS scenarios. The SON Autonomic Layer is implemented by a combination of custom and well-known opensource tools matching the modular design.

Data Collection is carried out at application-level by a Python client-server application composed by the Raw Data Pollster Agent (RDPA) running on each instance (client-side), and the Central Collector Node (CCN) on the server-side. They are communicated by means of HTTP REST messages pushed by the RDPA on regular time-intervals. Apart from that, the virtual infrastructure monitoring is natively supported by Openstack Ceilometer which gathers an extensive set of meters [Cl1] related to the cloud deployment. Depending on the type of measurement, notification or polling methods are implemented.

Data Aggregation is performed by two Python modules. On the one hand, the Feature

Extraction (FE) module fetches from the CCN a batch of client-request processing times observed on one-second intervals. Then, the FE computes the Rènyi entropy of the sampled observations, thus creating a time series of measurements. On the other hand, the Virtual Resource Aggregator (VRA) queries the Ceilometer REST API and extracts CPU usage counters for each analyzed virtual instance on one-second intervals, thus leading also to the creation of a time series.

EDoS detection is implemented on two major functional blocks: Analytics and Decision-Making (DM). The EDoS Analysis (EA) is a Java module implementing the Double Exponential Smoothing algorithm which results are stored in data structures, and the forecasted metrics are taken into account to elaborate the adaptive thresholds. In addition, EA implements the Clustering capabilities of the Weka libraries [Wkl], and the Decision Maker (DM) is implemented in Drools [SDB] as a rule-based expert system whose conclusions (alerts) are written in JSON files to match the Notification stage.

Finally, the client-side of the experimentation is performed by Python multi-threading modules capable to emulate a configurable number of clients. It also allows the possibility to control the number of requests and their connection rate to generate network traffic adapted to the experimental W-EDoS and I-EDoS scenarios.

8.7.2 Evaluation Methodology

The methodology used to conduct the experimentation is explained in the forthcoming sections, which distinguishes tests for assessing the W-EDoS and I-EDoS recognition capabilities of the proposal.

8.7.2.1 W-EDoS detection evaluation

The evaluation of W-EDoS analyzes the Rènyi entropy degrees measured on the server-side response times observed in one-second intervals, compared against the CPU consumption at operating-system-level. To this end, a client-server application has been implemented as described below.

- *Server-side software.* A RESTful HTTP application was written in Flask [SFl] due to the simplicity that entails this type of web service, hence exposing eight endpoints with different computational costs each in terms of processing time. When requested, each of them performs list-sorting operations with randomly generated values, varying on the list size and the number of iterations. The HTTP endpoints and their average execution times measured for 1000 executions are described in Table 8.3. All the HTTP methods support a GET parameter (id) used to distinguish client requests when they are generated from the same host (thus with the same IP). This application is hosted in a server deployed as an Openstack Compute instance running on a single-tenant private cloud.

- *Client-side software.* A Python application instantiates each client as a standalone thread, which in turn send HTTP requests to the web service following a random Poisson distribution being lambda (λ) the expected number of occurrences in a given

time interval. Such distribution is suitable to generate legitimate web requests as often assumed by the bibliography [BTI+03]. Thereby, normal traffic conditions and EDoS attacks can be conducted with multiple clients running in parallel. Taking advantage of the HTTP GET id parameter of each endpoint, a single node can impersonate multiple clients. For example, two HTTP requests launched from the IP 192.168.5.48 with URIs /3?id=145 and /3?id=213 would be interpreted as originated from different clients with ids 145 and 213 respectively. So that, response times measurements on the server are associated to those client ids.

- *Cloud Auto-scaling.* It has been configured an auto-scaling policy for launching a new virtual instance of the web service when the average CPU consumption runs above 60% in a one-minute interval.

- *W-EDoS attack scenarios.* The detection of a W-EDoS attack is carried out by examining the variation of the threat features in terms of the number of clients, request rates and percentage of malicious connections triggered from compromised nodes. Normal traffic conditions were considered when clients launched requests to endpoints 1 to 8, randomly chosen for each connection; whereas W-EDoS attack traffic is targeted to execute only endpoint 8 since it produces the costliest operations at server-side and the maximum CPU overhead in consequence. Table 8.4 summarizes the parameters considered to define the W-EDoS scenarios for normal network traffic, and Table 8.5 shows the malicious traffic volume determined by the attack intensity on each scenario (S1 to S4).

- *SON W-EDoS detection and notification.* The SON layer follows the principles of Definition 8.5.1. If a significative variation between the observed and forecasted entropy degree of the request processing times ($H_\alpha(X_{app})$) matches with an increment of the CPU consumption at operating system-level (X_{app}), the autonomic layer infers a W-EDoS attack in the analyzed web server and triggers an alert intended to prevent the auto-scaling of a new virtual instance since it does not resemble a legitimate origin.

Table 8.3: HTTP endpoints and average CPU processing time

URI	**Avg. CPU time (ms)**
/1	18.56
/2	21.58
/3	24.64
/4	27.81
/5	31.06
/6	33.72
/7	36.73
/8	226.04

Table 8.4: Normal traffic attributes

Feature	S1	S2	S3	S4
Number of clients (C)	1000	1000	1000	1000
λ requests per second (px)	50	60	70	80
Number of requests (R)	9000	10800	12600	14400

Table 8.5: Malicious requests per scenario

Intensity	S1	S2	S3	S4
1%	90	108	126	144
5%	450	540	630	720
10%	900	1080	1260	1440

8.7.2.2 I-EDoS detection evaluation

The evaluation of I-EDoS aims to evaluate inconsistencies between the number of deployed VNF instances compared against the productivity measured on them. This is implemented as follows:

- *VNF and productivity.* A simple Flask REST web server with a single endpoint of average response-time of 27.89 ms has been implemented and deployed as an Ubuntu-based Openstack Glance image, used as a template for launching additional instances in the cluster when scaling-out is performed. The performance of this VNF has individually evaluated with Httperf [THt], a well-known benchmarking open-source tool to measure web performance under different workload conditions whose results are shown in Table 8.6. Therefore, the ideal performance is reached by Medium and Optimal productivity levels at any time of operation, which are analyzed individually on each virtual instance.

- *Cloud environment configuration.* To showcase the evaluation scenarios, an Openstack cluster has been deployed and auto-scaling rules have also been configured to dynamically scale-out and scale-in the cluster size. It has been considered a minimum size of 2 instances and a maximum size of 12. A Neutron load balancer has also been deployed, which implements a round-robin policy to handle the traffic request to be distributed across the active cluster nodes.

- *Cloud auto-scaling in and out policy.* To scale-out the cluster, a new VNF instance is launched when the cluster average CPU consumption runs above 80% in a one-minute interval. Likewise, when the average CPU usage runs below 40% a virtual machine instance is removed, following the "youngest first" deletion policy. In addition, a "best effort scaling" is enabled to prevent breaking the size limitations (minimum and maximum) of the cluster.

- *Normal traffic conditions.* A normal scenario has been defined by modeling a traffic profile based on a variable number of expected HTTP connection rate (λ), measured in requests per second (px), in different time slots, elapsing a total time-window

of three hours. The cloud platform will then determine the right number of VNF instances based on the auto-scaling rules, ranging from 2 to 12 according to the cluster configuration. A graphical representation of the traffic workload is illustrated in Figure 8.11.

- *I-EDoS attack scenarios.* Auto-scaling is configured in Openstack Senlin, which is also integrated with Heat and Ceilometer to define cluster-related configuration templates with auto-scaling policies. Ceilometer alarms are thereby configured to trigger notification when some virtual-infrastructure-related counters run outside pre-defined thresholds, such as with CPU or memory usage. Based on the dependance on Ceilometer measurements, the I-EDoS attack is conducted by poisoning the counters gathered by Ceilometer related with CPU consumption since scaling policies rely on their values. To this end, it is assumed the ability of the attacker to gain access to the RabbitMQ message broker, either with legitimate authentication credentials, such as a valid username and password, or by exploiting a common Telemetry-related vulnerability, such as the one reported in CVE-2016-9877 [Cv9] where access to the messaging service can be granted only with a legitimate user regardless of the provided password. Once connected to the message bus, the attacker fetches legitimate CPU usage counters obtained by Ceilometer (represented as JSON objects) and injects a manipulated version of the original message with random CPU usage counters above 90 percent. As a result, and maintaining the same normal traffic conditions described in the previous section, the Heat engine performs scaling-out decisions based on unrealistic data, creating in the meantime more instances (cluster nodes) than needed to process the traffic workload.

- *SON I-EDoS detection and notification.* The SON autonomic layer forecasts the number of virtual instances in the Openstack cluster with the Double Exponential Smoothing (DES) algorithm. Meanwhile, the adaptive thresholds are built to look for situations where the number of existing instances is higher than the predicted values. The EDoS detection module targets these situations of unexpected growth to carry out DBSCAN density-based clustering based on the productivity measured on each virtual instance. It leads to the distinction between productive and lazy instances, which corresponds with Definition 8.6.3, triggering in consequence an instantiation-based unexpected behavior alert.

8.8 Results

The effectiveness of the proposal when analyzing EDoS threats and legitimate situations are described and discussed below.

8.8.1 W-EDoS detection

As indicated in Section 8.7, the evaluation of the W-EDoS detector is based on studying their behavior when varying the features of the threats and the legitimate environment,

Table 8.6: VNF performance and metrics

Connection rate (cons/sec)	Reply time (ms)	Reply rate (replies/sec)	Connection errors (%)	Avg. server CPU (%)	Productivity Label
5	31.9	5	0	22.9	Lazy
10	30.7	10	0	39.1	
15	29.5	15	0	49.3	Medium
20	27.2	20	0	57.5	
25	25.9	25	0	66.1	Optimal
30	25.3	30	0	74.3	
35	28.2	35	0	81.6	
40	133.1	38.2	4	92.1	Overload
45	736.2	8.1	89	99.6	

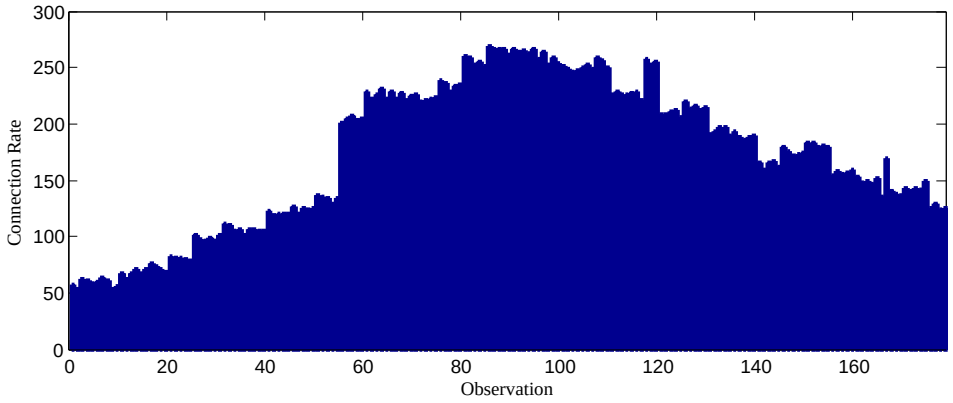


Figure 8.11: Traffic workload for I-EDoS experimentation.

the impact of the Rènyi entropy degree and the improvement obtained by refining the analysis of the X_{App} evolution with the study of X_{CPU}. They are described throughout the rest of this subsection.

8.8.1.1 Attack distribution and requests

The results obtained when varying the request rate and the intensity of the W-EDoS threats are summarized in Figure 8.12(a). With the purpose of facilitate their understanding, the intensity of attacks has been measured at the interval 1%, 5%, 10%, where the percentage indicates the distribution of malicious requests per observation. In addition, four different scenarios have been studied based on the number of average requests per second (px) made by each client: $\{50, 60, 70, 80\}$. In Figs. 8.12(a)-8.12(c) the ROC curve obtained at each experiment is illustrated, where it is assumed the K value of the prediction interval as the main parameter that adjusts the W-EDoS detection sensitivity. The obtained effectiveness is summarized in Table 8.7 and Figure 8.12(e). The worst results are observed when clients perform an average of 60 requests per second, as well as when the attack is present in at least 1% of the request, being the trapezoidal approximation to the Area Under the Curve (AUC) 0.901, and in the best case under this setting, the True Positive Rate (TPR) 0.816 and the False Positive Rate (FPR) 0.15. On

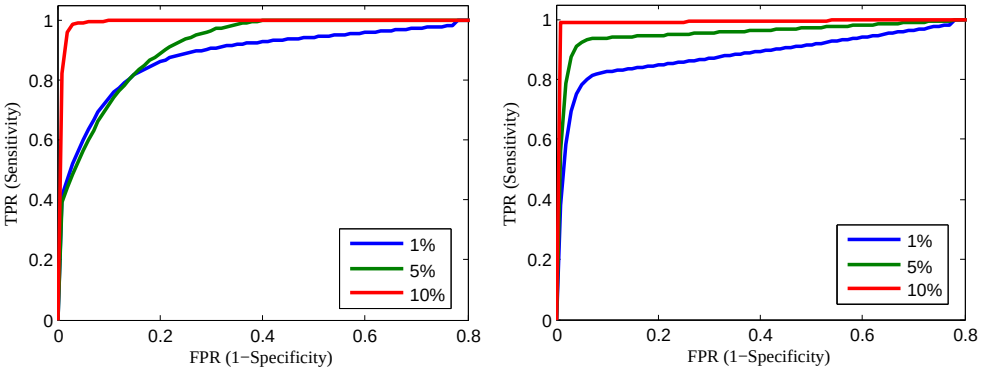
the opposite, the W-EDoS detector best performs when the sensors submit an average of at least 80 requests per sensor and the attack poses 10% intensity. In this case the AUC is 0.995, and the best TPR is 1 and FPR is 0.01. From the plots and the summarized data, it is possible to conclude that, as the number of requests per node grows, the accuracy of the system improves. This denotes that under such circumstance, the prediction strategy has a greater capacity to model the characteristics of the monitored traffic, and therefore to act more accurately. On the other hand, it is possible to affirm that the greater attack intensity, the higher its visibility, resulting in much more obvious entropy variations and CPU overload. In general terms, the obtained accuracy demonstrates the ability of the W-EDoS detection method of operating on scenarios similar to those proposed in the testbed.

Table 8.7: Summary of results of the W-EDoS detector

	Average Requests per Second (px)											
	50px			60px			70px			80px		
Attack	AUC	TP	FP	AUC	TP	FP	AUC	TP	FP	AUC	TP	FP
1%	0.9069	0.8183	0.08	0.9013	0.8161	0.15	0.9175	0.83654474	0.07	0.9345	0.8492	0.05
5%	0.9557	0.9113	0.04	0.9432	0.8195	0.15	0.9631	0.9347	0.05	0.9603	0.9434	0.11
10%	0.991	0.9889	0.01	0.9631	0.9843	0.02	0.9908	0.989	0.01	0.995	1	0.01

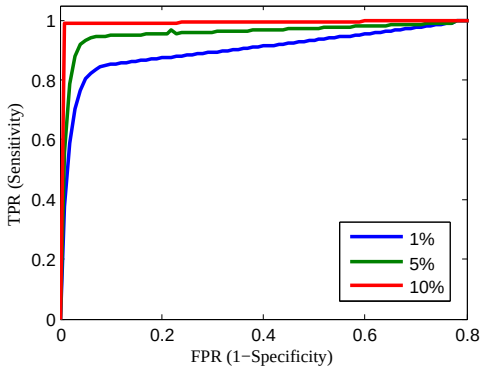
8.8.1.2 Entropy degree

In the performed experimentation, the Rènyi entropy degree that configures the W-EDoS detectors has proven to play an essential role in the quality of the decisions made. With evaluation purposes, and in view of the results observed in subsection 8.8.1.1, a testing scenario with average rate of 60 requests per client/second, and attacks originated from 5% of the clients to be served, has been considered. It is worth mentioning that this configuration represents an intermediate point between the circumstances that provide greater visibility of the threats, and those that hinder their identification. The results obtained are illustrated in Figure 8.13, where it is possible to distinguish the impact of α in the ROC space (Figure 8.13(a)), the summary of the precision obtained in terms of TPR and FPR (Figure 8.13(b)), and an example of its smoothing effect in the time series to be analyzed (Figure 8.13(c)). Note that at this study the range $1 \leq \ alpha \leq 5$ has been studied. The minimum value supported by the Rènyi entropy is $\alpha = 0$, which has not been taken into account since it coincides with the expression of Hartley (i.e. max-entropy) $H_\alpha(X) = \log_2 n$, which due that normalized values are adopted, always returns 1. From $\alpha = 5$, the observed AUC is less than 0.5, so it is inferred that the obtained values correspond to a random behavior. From Figure 8.13(a) and Table 8.8 it is possible deduce that the best fit is $\alpha = 1$, where AUC = 0.955, TPR = 0.911 and FPR = 0.04. Hence, as the degree increases, the smoothing of the resulting time series decreases, which entails a greater tendency to report false positives, (see Figure 8.13(c)). Because of this, the worst measured value has been observed when $\alpha = 5$, resulting in AUC=0.513, TPR=0.556 and FPR=0.64, which obviously are far from achieving a desired accuracy. Consequently, it is possible to conclude that the degree of Rènyi entropy has a direct relationship with the successfulness of the performed deployment; and that in order to mitigate the false

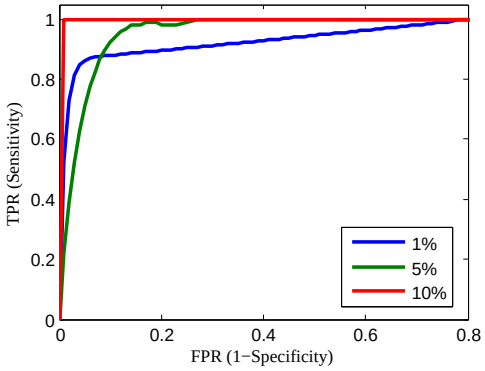


(a) 50px

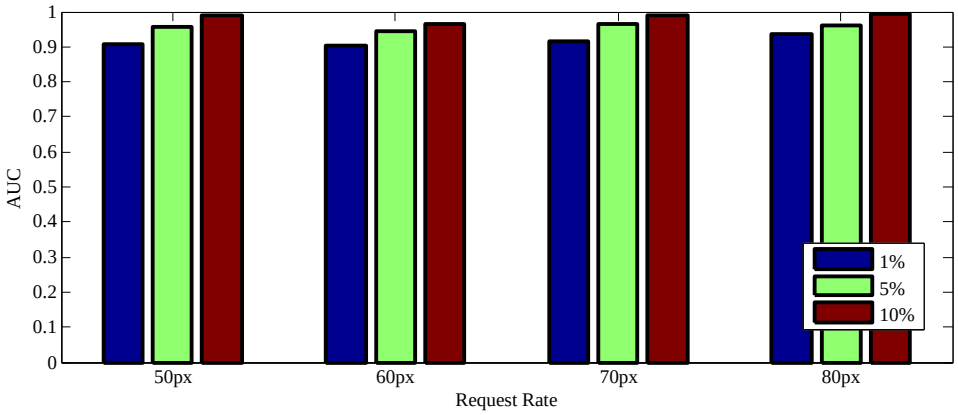
(b) 60px



(c) 70px



(d) 80px



(e) AUC Comparison

Figure 8.12: Results when varying the attack rate and requests

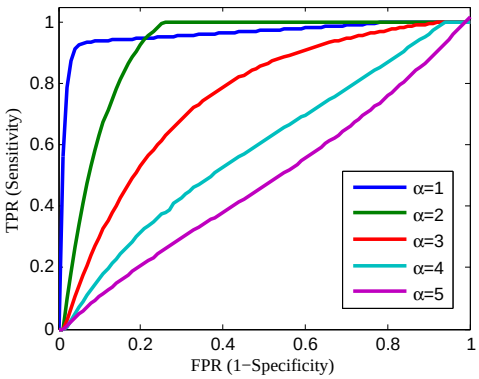
positives problem, it is advisable to select low values, thus reducing the impact of the temporary outliers inherent to the monitoring tasks on network environments.

8.8.1.3 Simple X_{App} and refinement by X_{CPU}

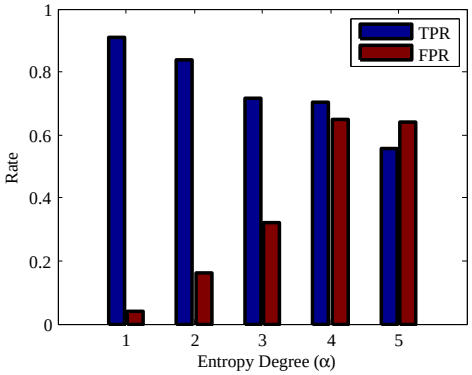
In order to evaluate the refinement of the analysis stage based solely on X_{App}, achieved by studiying X_{CPU}, a new test has been performed. Firstly, it only attempted to detect W-EDoS based on X_{App}, then assuming the combination of both random variables. The

Table 8.8: Summary of results when varying α

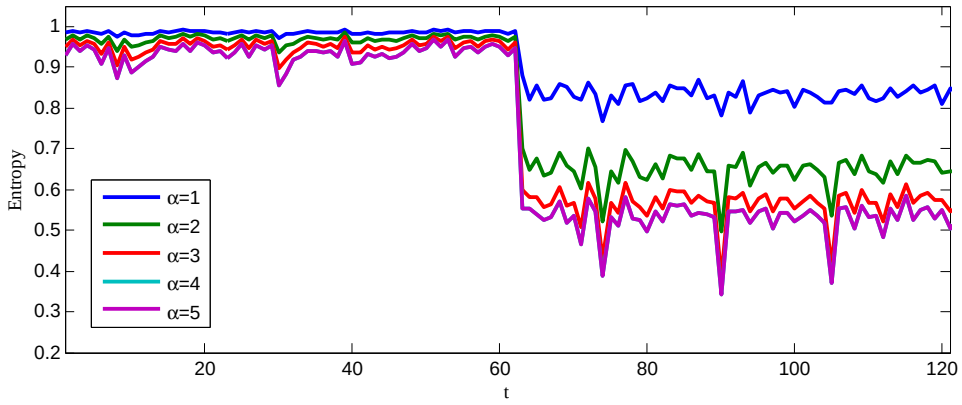
	Rènyi entropy degree				
Measure	**α=1**	**α=2**	**α=3**	**α=4**	**α=5**
AUC	0.9557	0.8692	0.7574	0.6072	0.5138
TPR	0.9113	0.8388	0.7147	0.7041	0.5568
FPR	0.0400	0.1600	0.3200	0.6500	0.6400



(a) ROC varying α



(b) Effectiveness



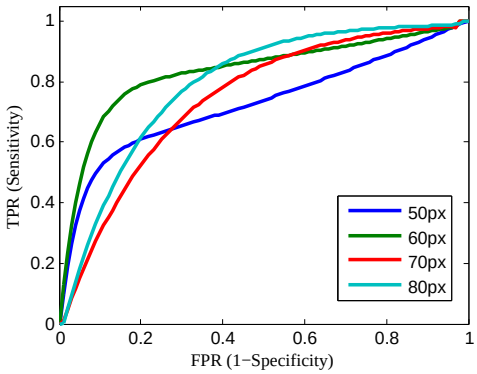
(c) Variations on Entropy degree (α)

Figure 8.13: Analysis of entropy degree variation

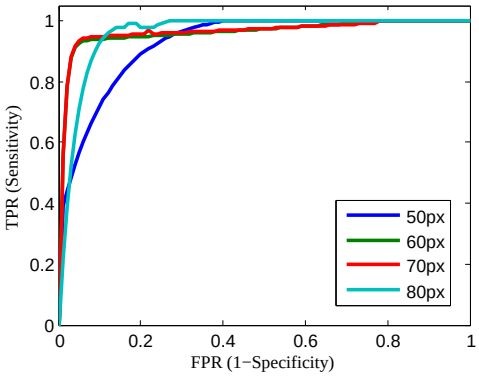
obtained effectiveness has been assessed based on the number of average requests per second (px) made per client: $\{50, 60, 70, 80\}$, and by configuring 5% of the clients to be served as attackers. The results are illustrated in Figure 8.14, where Figure 8.14(a) indicates the precision obtained to consider only X_{App}; Figure 8.14(b) displays the results when combining both random variables; and Figure 8.14(c) plots its comparison, which contents are summarized in Table 8.9. The best results in the simple X_{App} tests were AUC=0.7766, TPR=0.5152 and FPR=0.1, when 80px. On the opposite, the combined setting provided AUC=0.9633, TPR=0.9346 and FPR=0.05 when 70px. Note that as discussed in Section 8.8.1.1, the greater numbers of average requests per second tend to facilitate the detection task. In the light of these results, it is possible to deduce that the W-EDoS detection when only considering X_{App} was not feasible, since the AUC fall

Table 8.9: Summary of results considering X_{App} and X_{CPU}

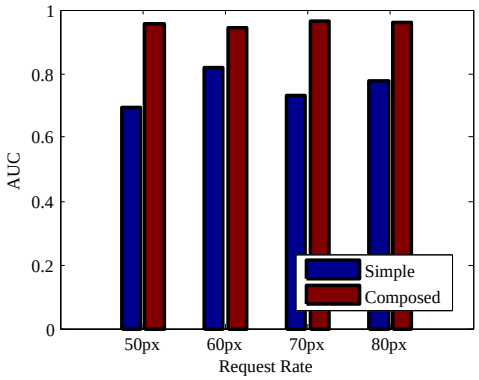
		Effectiveness measurements		
Setting	**Px**	**AUC**	**TPR**	**FPR**
X_{App}	50	0.6942	0.5152	0.10
	60	0.8189	0.7028	0.12
	70	0.7319	0.7021	0.32
	80	0.7766	0.7541	0.29
X_{App} and X_{CPU}	50	0.9557	0.8195	0.15
	60	0.9432	0.9113	0.04
	70	0.9633	0.9346	0.05
	80	0.9603	0.9434	0.11



(a) Effectiveness when only analyzing X_{App}



(b) Effectiveness when analyzing X_{App} and X_{CPU}



(c) AUC Comparison

Figure 8.14: Analysis when considering X_{App} and X_{CPU}

behind those provided by the combined approach, as well as an alarming false positive rate was displayed. This is because of the situations previously described in Section 8.5.2, where it was explained the rebound effect of the forecasting modes, and other legitimate situations that may lead to false positives.

8.8.2 I-EDoS detection

This section describes the results observed when evaluating the I-EDoS detection capabilities of the proposed defensive scheme. In order to facilitate the understanding of the performed research, an illustrative case of study is detailed, where the differences between normal activities and I-EDoS are displayed, and the detection process is demonstrated. In addition, the effectiveness of the approach when dealing with attacks of different intensity is discussed.

8.8.2.1 Case of study

The testing scenario evaluates the effectiveness of the I-EDoS detection in the experimental testbed described in Section 8.7. The comparison between the number of VNF instances deployed as a result of auto-scaling decisions on both normal and attack traffic are presented in Figure 8.15, where it is noticeable that the deployment of a higher number of VNF instances has been caused by an I-EDoS attack. Under normal conditions, the cluster scaled up to a maximum of 10 instances when the highest traffic workload has been reached; whereas the I-EDoS attack has forced the scaling of the cluster up to the maximum size of 12 instances. Then, the number of virtual instances have been compared with the adaptive thresholds estimated for normal (Figure 8.16(a)) and attack conditions (Figure 8.16(b)) in which deviations between the forecasting and the real measurements have been found. For instance; at observations 12 or 36 when analyzing normal traffic, and at observations 14 or 34 when the attack was performed. Besides the identification of a suspicious growth of the VNF instances, the I-EDoS detection strategy relies on the productivity measurements, whose dispersion graphs are compared. Under normal coniditions (Figure 8.17(a)), the lower number of instances leads to a more uniform distribution of the productivity exposed by the VNFs which in general reach higher values and less dispersed measurements than its counterpart when the I-EDoS attack was launched (Figure 8.17(b)). In such attack conditions, the red dots clearly denote the presence of a subset of instances whose productivity is significantly lower, thus being referred as lazy nodes. In consequence, the overall productivity is decremented since this scenario forces the workload distribution into a higher number of VNF instances. To quantitively validate the distinction of productive and lazy instances, the results of the DBSCAN density-based clustering are presented for sampled measurements obtained at t=86 under normal traffic conditions (Figure 8.18(a)). Three gropus of instances are created (Group A, Group B and lazy), with 80% of instances allocated to Groups A and B and remaining of 20% allocated to the group of possible lazy instances. Likewise; taking the productivity measurements observed at t=19 when the attack is performed (Figure 8.18(b)), DBSCAN generates two groups. The lazy group gathers 73% of the allocated instances, while the latter poses the remaining 27% as productive instances.

8.8.2.2 Attack Intensity

As would seem logical, the intensity of the I-EDoS attacks has directly influenced the detection capability of the approach. Figure 8.19(a) and Table 8.10 display the

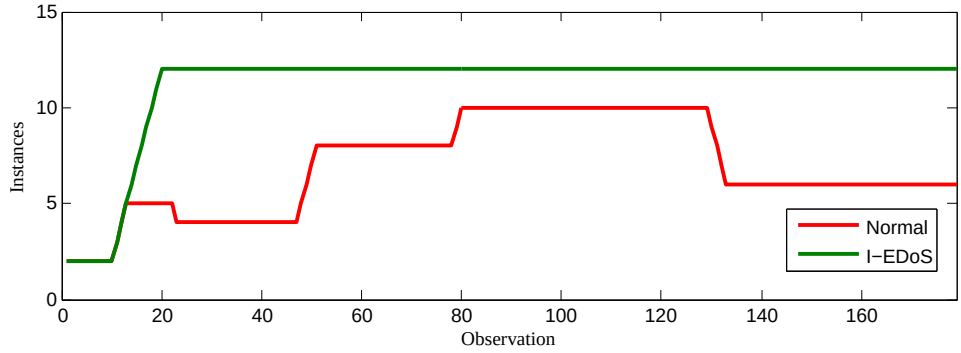
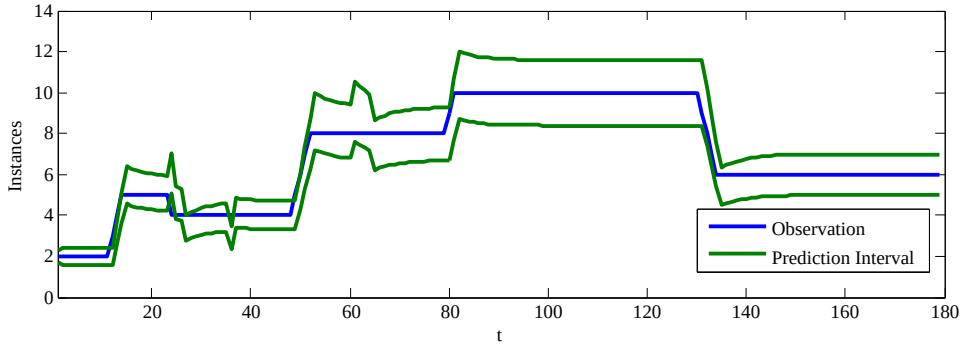
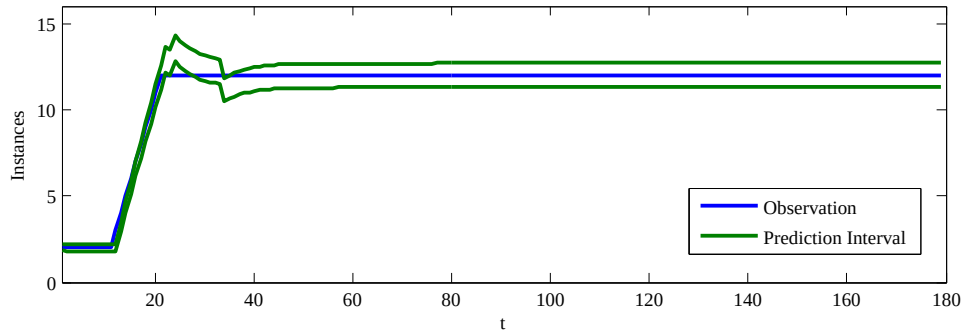


Figure 8.15: Number of instances deployed in normal and attack scenarios.



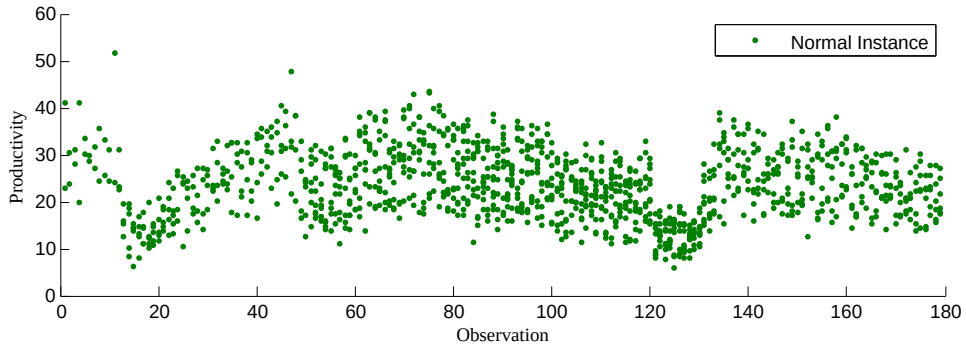
(a) Normal traffic



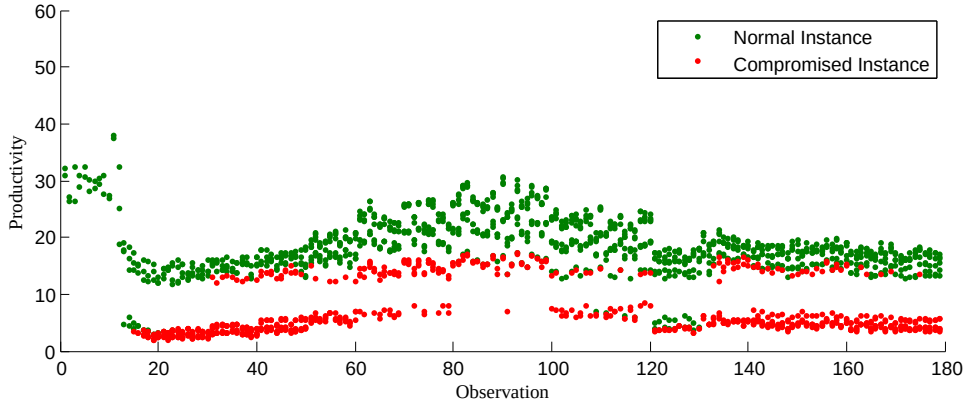
(b) I-EDoS attack

Figure 8.16: Forecasting of the number of VNF instances

effectiveness registered when analyzing threats that lead to increase 10%, 20%, 30% 40% and 50% the number of instances of the NFV considered at the previous example. In general terms, the success rate has varied little (see Figure 8.19(c)); at the first four groups of attacks (10%, 20%, 30%, 40%) a distance of 0.022 (0.025%) was observed between the minimum hit rate (TPR=0.89, in 10%) and the best hit rate (TPR=0.91 in 40%). When the attacks posed higher intensity (50%) the hit rate slightly improved (TPR=0.94 in 50%). However, by taking into account the issuing of false positives the results were more significant (see Figure 8.19(d)); in particular, if the attacks posed lower intensity, the detection method reached FPR=0.12; but by increasing their capacity to cause economic losses, the best observation decreased to FPR = 0.07, which represents

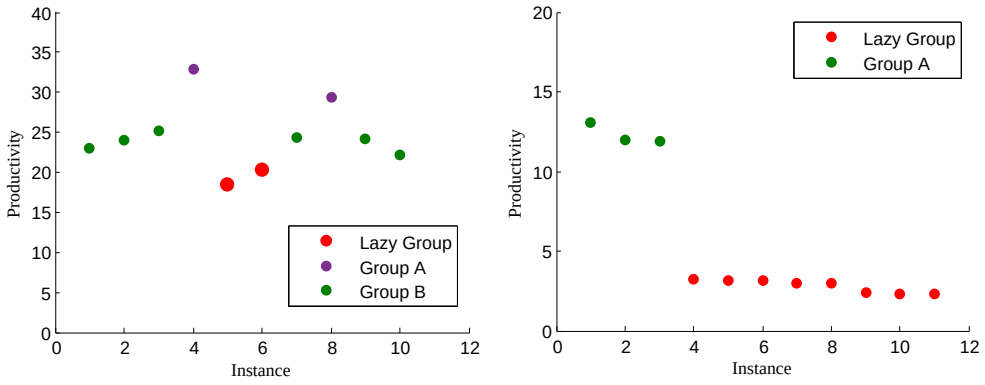


(a) Normal traffic



(b) I-EDoS attack

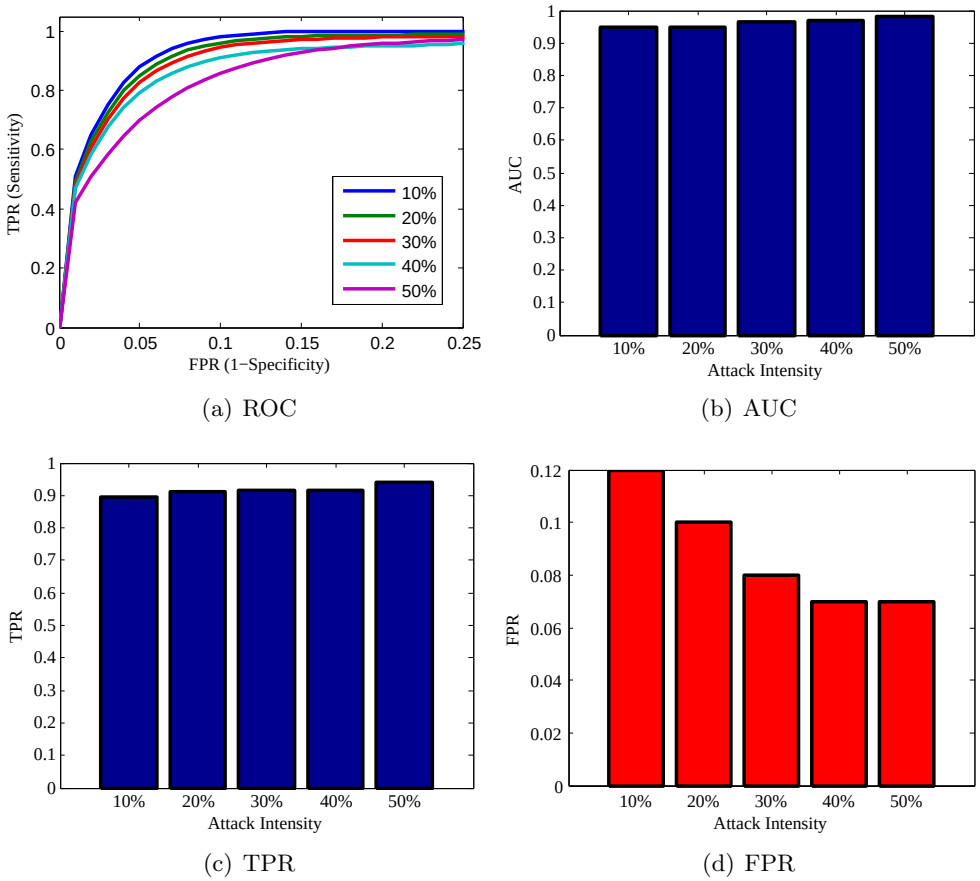
Figure 8.17: Productivity distribution per VNF instance



(a) Normal traffic classification at t=86 (b) Attack traffic classification at t=19

Figure 8.18: DBSCAN classification by VNF productivity

an improvement of 58.3% over the worst registration. These situations are reflected in Figure 8.19(a) and Figure 8.19(b), where the AUC varies according to the attack intensity, being AUC=0.9483 in the worst case, and AUC=0.9811 in the best case. The effectiveness variations are mainly due to the clustering stage, where the instantiated NFVs are grouped based on productivity. The greater is the attack intensity, the greater is the number of instances created by the intruded that belong to the lazy group. This places a larger



(a) ROC (b) AUC

(c) TPR (d) FPR

Figure 8.19: Analysis of the effectiveness at different attack intensities

amount of normal instances in other groups, hence reducing the false positive rate in the best calibrations despite generally, it remains similar. In view of the results, it is possible to conclude that the proposed strategy is capable of detecting I-EDoS threats successfully. However, to maintain a setting that facilitates recognition of low intensity threats may result in the emission of a greater number of false positives, which should be considered by the security management strategy. It entails deciding the trade-off between protection and economic losses that fits better into the monitoring scenario and the security policies.

Table 8.10: Summary of results considering different attack intensities

		Effectiveness measurements	
Intensity (%)	**AUC**	**TPR**	**FPR**
10	0.9483	0.8936	0.12
20	0.9498	0.9099	0.10
30	0.9654	0.9167	0.08
40	0.9708	0.9155	0.07
50	0.9811	0.9404	0.07

Bearing the performed experimentation in mind, the following items are highlighted:

1. When analyzing W-EDoS, as the number of requests per node and the intensity of the attack increase, the accuracy of the system improves. The best results were observed when the sensors submit an average of at least 80 requests per sensor and the attack poses 10% intensity (AUC=0.995, TPR=1, FPR=0.01).

2. The W-EDoS detection approach is more effective when the Rènyi entropy degree is low. The best observation was $\alpha = 1$ (AUC=0.9557, TPR=0.9113, FPR=0.0400) with a test setting that assumed an average rate of 60 requests per client/second, and attacks originated from 5% of the clients to be served

3. The W-EDoS detection when only considering X_{App} was not recommended because of the rebound effect of the forecasting and adaptive thresholding methods described in Section 8.5.1. It leads to a significant increase in the false positive rate.

4. The proposal for I-EDoS detection behaves better when analyzing attacks of greater intensity. The best results were observed when the attack intensity was 50% (AUC=0.9811, TPR=0.9404, FPR=0.07).

5. At I-EDoS detection, the greater is the attack intensity, the greater is the number of instances created by the intruded that belong to the lazy group. This has a direct impact on the false positive rate resultant of the optimal calibration. Hence if the attacks posed lower intensity, the detection method reached FPR=0.12; but by increasing their intensity, the best observation decreased to FPR = 0.07, which represents an improvement of 58.3% over the worst register

From these facts it is deduced that the main objective of this proposal has been achieved, i.e. it was able to successfully recognize the different EDoS attacks against the self-organized network considered during the experimentation. However, the effectiveness has varied depending on different characteristics of the monitoring scenario, highlighting among them the heterogeneity of the information to be studied (1) and the intensity of the attack (1), (2), (4), (5). The first one allows more accurate modeling of the legitimate behavior by reducing the false positive rate, and the second facilitates the recognition of the intrusive activities. Finally, it is worth highlighting the impact of the Rènyi entropy degree at the W-EDoS detection approach (3), where this parameter significantly influenced the smoothing and noise reduction of the time series to be analyzed. This has resulted in the fact that $\alpha = 1$ (Shannon entropy), i.e. the most noise-tolerant setting, entailed greater accuracy, in a similar way to previous publications related with conventional DDoS recognition [ÖB15].

From an analytical point of view, a light solution has been proposed, where the most complex tasks are executed in a dedicated server, thus minimizing operational costs. Given that the definitions of W-EDoS and I-EDoS attacks lie on CRoWN indicators, the object of study has been clearly delimited. But certainly, the future will bring new ways of achieving EDoS, so it is not possible to guarantee the effectiveness of the proposal in uncharted circumstances. It is also important to bear in mind that the proposed

solution implements analytic tools well-known by the research community, for example the DES [GJD80] forecast algorithm or the DBSCAN [EKS+96] clustering method. Their selection has been conveniently justified throughout the paper, but they pose advantages and disadvantages that should be considered prior to instantiation in alternative use cases. Note that if necessary, they can be replaced by similar algorithms. Finally, it should be highlighted that the proposal inherits the countermeasures related with the limitations assumed at the design principles (see Section 8.3). Therefore, it does not incorporate adaptation techniques to non-stationary scenarios [DRAP15] nor robustness against adversarial attacks [ÖB15, ADAH14]. Neither an advanced information correlation strategy has been proposed, where data from different sensors could be pooled. These facilitate the inference of more precise diagnoses [SMFDV13, MAJ13], which serve to risk assessment and to resolve the trade-off between the cost of countermeasures deployment and the estimated losses caused by the intrusion. Consequently, many interesting lines of future work have been raised.

8.9 Final Remarks

This chapter has delved into the analysis of EDoS threats in emergent self-organized networks. In order to lay a formal definition of EDoS threats, a set of indicators have been identified that lead to the distinction of two categories of EDoS threats: Workload-based EDoS and Instantiation-based EDoS. They constitute one of the central contributions of this work on which the defensive detections strategies have been conducted. The W-EDoS attack detection considers significant prediction errors terms of CPU consumption and response-time at application-level of the instantiated VNFs. On the other hand, I-EDoS attack detection analyzes the relationship between the initialization of unproductive instances and the suspicious growth of the number of deployed NFVs instances. To validate the proposal an extensive experimentation has been conducted in a self-organized testbed, and the obtained results have proven good accuracy on the detection of such threats. Therefore, it is possible to conclude that the proposal fulfills its main objective at the deployed scenario.

Chapter 9

Detecting the Participation of a Device in a DDoS Attack

This chapter introduces a novel approach for detecting the participation of a protected network device in Distributed Denial of Service attacks. With this purpose, the traffic flows are inspected at source-side looking for discordant behaviors. To this end, the strategy has led to delegate the analytic tasks to a dedicated autonomic layer, hence minimizing the impact on operational efficiency and quality of experience. In particular, the approach takes advantage of the knowledge acquisition framework implemented in the SELFNET project, which facilitates its implementation as a self-organizing solution. The proposal adopts feature extraction, pattern recognition, prediction and adaptive thresholding capabilities, and facilitates its adaptation to more sophisticated self-protection approaches. Hence, the main contributions of the performed research are:

- An in-depth review of the flooding-based DDoS landscape and the different proposals for its mitigation from the academic point of view.
- The introduction of a novel method for its identification by analyzing source-side activities, adapted to non-stationary of the emerging networks.
- An adaptation of the solution to an advanced architecture for self-protective purposes.
- The collection of a dataset.
- The description of an evaluation methodology for proving the effectiveness of the proposal.
- A comprehensively discussion of the obtained results.

This chapter is structured into seven sections. In Section 9.1, the initial considerations for the proposal are described categorized as design principles, assumptions and limitations. Section 9.2 defines a multi-layered architecture on which the detection strategy is implemented. Section 9.3 provides a description of the DDoS indicators considered in the detection stage. Section 9.4 describes in detail the detection strategy composed by three

major data processing stages. In Section 9.5 the evaluation methodology is presented. In Section 9.6 the results derived from the experimentation are discussed. Finally, Section 9.7 remarks the conclusions of this chapter.

9.1 Initial Considerations

This section delves into the design principles reviewing its objectives, assumptions and limitations. It also describes the current architecture and the strategy for acquiring initial factual knowledge, in this way detailing the procedures for monitoring metric generation and knowledge inference.

9.1.1 Design principles

The defense against flooding-based DDoS attacks may be approached from different perspectives, ranging from prevention to identification of sources [VZF17]. In addition, and given the complexity of the emerging network scenarios, they pose a large number of challenges, as is the case of deciding the most effective countermeasures and range of action [ZJT13], to adapt to the nature of the data to be modeled [BBK15] or how implement previously agreed security management policies [Den14b]. In order to facilitate the understanding of the performed research, it should be clear that main objective of this contribution has been the development of a novel flooding-based DDoS attacks detection strategy at source-side, that must to adapt to non-stationary processes in the data to be analyzed. Unlike similar proposals, only a single source of information is monitored, which is the protected device [MKK11]. The most relevant secondary goals are the evaluation of the approach under different traffic profiles; and its integration as a self-organizing solution for next generation networks, thus allowing both taking advantage of its dedicated analytical capabilities and fostering the definition of sophisticated use cases able to reactively/proactively manage the protected network security. To this end, the SELFNET project [P5S] has been selected, which aligns with the European 5G PPP Security Work Group.

9.1.2 Assumptions

In order to restrict and lay the foundations of the performed research the following premises have been assumed:

- The detection of the participation of an end-user or IoT device as source-side of DDoS attacks based on the study of metrics aggregated from its incoming/outgoing traffic is possible.
- Flooding-based DoS differ from normal activities in traits related with number of requests observed and traffic volume generated by the suspicious end-points. In the case of DDoS attacks the number of clients involved also varies [SMMVGV17a].
- The analysis of discordant behaviors in aggregated metrics at flow-level enables recognizing DDoS situations on conventional monitoring scenarios [ÖB15].

- By extracting and analyzing advanced metrics in a remote dedicated server it is possible to provide a passive-monitoring detection approach.
- As the information provided by incoming/outgoing traffic flows monitored from network devices largely depends on the traffic profile, its non-stationarity is assumed (since for not all users this feature can be guaranteed). The non-stationarity is also inherent to the emergent communication networks landscape.

9.1.3 Limitations

For diverse reasons, the performed research has not taken into account the following circumstances, most of them being postponed for future work:

- The protection of communication channels between monitoring agents and SON analytical components has not been addressed [LII+15]. Security practices on software development are not explicitely considered either. Consequently, at the performed experimentation it was assumed they have not being compromised.
- Nowadays there are different adversarial threats able to evade detection methods similar to those studied during the course of the performed research [ÖB15]. But given the complexity that their development often entails and aiming on facilitating the understanding of the main contribution of our research, their adoption is out of the scope of this publication.
- The issues related with data protection inherent in the audition of user behaviors at communication networks have not been considered. Neither the implementation of the recent European General Data Protection Regulation (GDPR). Consequently, it is assumed that the proposed solution have permission to monitor the incoming/outgoing traffic of their network devices for purely analytic purposes.
- The knowledge representation and data models implemented for management and storage of the factual knowledge acquired by this proposal are not detailly specified throughout this work.

9.2 Architecture

The proposed SON architecture (Figure 9.1) is grounded in the functional layers defined for the SELFNET framework [P5S]. At a glance, this 5G oriented architecture takes advantage of the decoupling of the control and data plane layers, promoted by SDN to allow a fully software-driven management model, being this a remarkable characteristic of the next-generation networks [ARS16].

The major benefit of this model is the inclusion of complex data processing tasks in the SON Autonomic Layer, which encompass advanced data extraction features, machine learning approaches, anomaly detection strategies, among others, towards the accomplishment of self-protection capabilities for detecting and mitigating network threats in complex network contexts [MGC+18]. In the lowest level of this 5G-oriented

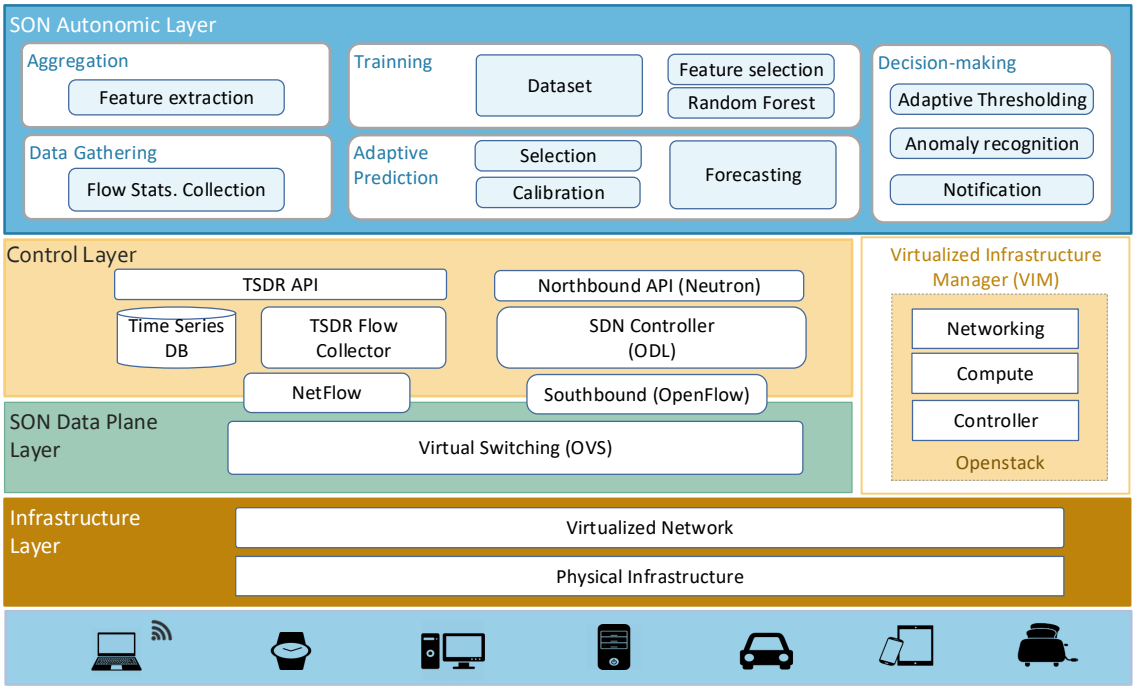


Figure 9.1: Architecture for Source-side DDoS Detection.

architecture, the network physical infrastructure holds heterogeneous network nodes embracing end-user devices (D2D), such as mobile phones and personal computers, or IoT devices intended for machine to machine (M2M) communications [PDG+16]. All of them act as traffic generators, thus increasing the complexity of the monitored environment as the network grows. On the other hand, the inclusion of virtualization capabilities driven by the Network Function Virtualization (NFV) architecture leads to overcome scalability issues of physical infrastructures [HH15] due to its on-demand provisioning model orchestrated by the Virtual Infrastructure Manager (implemented on Openstack [Ope]), which allows the instantiation of virtualized network elements easily configurable by software. This is a distinguishable aspect of 5G network architectures, which is achieved, for instance, by provisioning a VNF node with the desired protocol stack [TKJ16]. Thereby, virtualization leads to the creation of configurable forwarding nodes in the SON Data Plane layer, which are in the meantime compatible with the SDN paradigm. For this reason, the Open vSwitch (OVS) [POv] instances implement the OpenFlow [MAB+08] protocol in the southbound interface for handling the configuration messages with the SDN controller.

In addition, OVS switches provide support for NetFlow, a well-known protocol used for monitoring flow traffic statistics [HCT+14] built upon the matching of source IP, destination IP, and protocol; which are analyzed in the autonomic layer. NetFlow implements flow-sampling methods which have faced some scalability issues [LMKY16] in large network deployments as other flow monitoring methods, hence remaining as a traffic engineering open challenge in research literature [ALW+14] [SWXH15] [YHSH17]. Despite its drawbacks, this proposal has opted for NetFlow to prioritize the flow metrics extraction to validate the detection model, which is grounded on accurate flow statistics

rather than addressing efficiency and scalability aspects. Notwithstanding, those issues have been mitigated by provisioning the OVS instances with larger allocation of memory and computing capacity aimed to enhance its performance. Both OpenFlow and Netflow interact with the OpenDaylight (ODL) [MVTG14] controller located in the Control Layer. The OpenFlow interface manages the configuration of the flow-tables for packet forwarding in the data plane, whereas the Netflow collector gathers flow counters from the virtual switches. The proposed detection strategy relies on the representation of flow metrics as time series, being this the reason why the ODL Time Series Data Repository (TSDR) [ODT] is deployed to transform flow statistics into a time series representation. On the highest level of the architecture, the SON Autonomic layer is composed by data processing modules targeted on the core detection strategy implementation, which spans from the data collection to the notification of network threats. Flow statistics are gathered by querying the time series database (through the TSDR API) under different granularity levels, and the resultant sample of metrics are aggregated by means of feature extraction methods (entropy measurement). Likewise, to conduct a machine learning approach a Training stage is considered, of which main outcome is the construction of a classification model fitted for selecting the proper prediction algorithm based on the time series features included in the reference dataset. This classification model is used in the Adaptive Prediction stage when the most accurate prediction algorithm is inferred from the time series characteristics extracted from the monitored observations. Once selected, it is calibrated for minimizing the forecasting error to enhance its accuracy. Then, the DDoS detection is carried out by constructing the adaptive thresholds estimated from the predicted values to detect anomalies when the observations are outside the prediction boundaries, thus generating DDoS alerts to be notified as the outcome of the SON self-protection approach.

9.3 DDoS indicators

Throughout the performed research different levels of information processing have been studied, which entailed the need for extracting very heterogeneous features that facilitate the analysis of the knowledge acquired from the monitored devices, that being analyzed as univariant time series. They are summarized in Table 9.1 and described throughout this section.

9.3.1 Time Series Features

The first analytic stage has focused on the extraction of traits that allow defining usage models adaptable to changes in the monitoring environment. For the automation of deciding the best suited modeling and prediction strategies, more than 100 metrics per time series sample were constructed by the tool TSFRESH, which has been developed under the iPRODICT [TF1] project. This collection takes into account from basic statistical attributes (peaks, maximum/minimum observations, mode, etc.) to correlation measures related with the evolution of the time series (white noise, trend, seasonality, autocorrelation

Table 9.1: Source-side DDoS detection indicators.

Cumulative	
Symbol	Description
nP_{in}	Incoming packets
nP_{out}	Outgoing packets
nB_{in}	Incoming bytes
nB_{out}	Outgoing bytes

Flow Disorder (H)	
Symbol	Description
$H(nP_{in})$	Entropy packets per incoming flow
$H(nP_{out})$	Entorpy packets per outoging flow
$H(nB_{in})$	Entropy bytes per incoming flow
$H(nB_{out})$	Entropy bytes per outgoing flow

Flow Divergence ($nMSE$)	
Symbol	Description
$nMSE(nP)$	Diff. Input and output packets
$nMSE(nB)$	Diff. Input and output bytes

coefficients, etc.). They were directly applied on the collection M3-Competition [Mak00] at the training stage of the system.

9.3.2 Basic Metrics

The incoming/outgoing traffic flows from the protected IoT device are monitored and structured in IPFIX format [HCT+14], according to which each traffic flow being a bunch of packets captured in certain time interval t that share the following properties: same source IP address, IP destination and protocol. The timeslots that delimit the traffic flows establish the granularity of the analytic tasks to be performed, in this way serving as adjustment parameters that configure the sensitivity level of the detection methods. For example, when the granularity is high, the information to be processed is hardly filtered or softened, since it is often acquired from less instances (packets). As a result, these observations are more likely to pose outliers or noise. However, when the granularity is too low, it is possible that the analytic tasks overlook relevant situations. The first of these scenarios results in a more restrictive adjustment, where the detection of threats is prioritized in opposition to the generation of false positives. In the second case, the quality of the user experience is prioritized at the expense of decreasing the level of protection offered. The following pair of measurements is taken per traffic flow: number of transferred packets nP and total amount of information transferred nB (bytes). From them the aggregated metrics described in the next subsection are inferred.

9.3.3 Aggregated Metrics

As suggested in the bibliography, the basic flow-level metrics are aggregated based on the relationship between outgoing and incoming traffic and their dispersion [GP01]. In the first case, the normalized Mean Squared Error (MSE) is considered, which is expressed as follows:

$$nMSE(X) = \frac{\frac{1}{n}\sum_{i=1}^{n}\left(x(a)_i - \hat{x}(b)_i\right)^2}{\sigma^2} \tag{9.1}$$

where X is the trait to be analyzed, n is the total number of traffic flows with paired

IP source and IP destination (i.e. the traffic incoming/outgoing traffic between a and b), $x(a)_i$ is the metric registered at the incoming traffic grouped at the flow a, and $x(b)_i$ is the metric registered at the outgoing traffic at b. A clear example is illustrated in the relationship $E_\tau(nP_{in},\ nP_{out})$ that describes the difference between incoming packets $X_{in}(a) = nP_{out}(b)$ and outgoing packets $X_{out}(a) = nP_{in}(b)$ captured at the time interval τ.

On the other hand, the disorder degree of the observations is measured based on the normalized entropy described by Shannon. This decision is supported by previous research works related with conventional DDoS recognition, which successfully approached similar problems in the same way [BBK15]. It has been hypothesized about this metric being also valid for detection at single source-side device monitorization. As is usual in the case the bibliography, the entropy implemented by this proposal is inferred from the following expression:

$$H(X) = \frac{-\sum_{i=1}^{n} p_i \log_a p_i}{\log_a n} \tag{9.2}$$

where n is the total number of monitored flows captured at the time interval τ, and $p_1, p_2, \ldots, p_n$ are the probabilities of the instances $x_1, x_2, \ldots, x_n$ of the random variable X, the latest constructed from basic flow-level metrics. Table 9.1 summarizes the DDoS indicators studied at the performed experimentation.

9.4 Source-side flooding-based DDoS detection

The proposed architecture bases its detection strategy on studying univariate time series built from aggregated metrics, which are deduced from both traffic monitored at the protected devices and collections of reference time series with training and validation purposes (at the performed experimentation, the M3-Competition [Mak00] dataset). To this end, three major data processing stages are distinguished: Training, Adaptive Prediction and Classification (see Figure 9.2). At Training stage, the criteria that facilitates deciding the predictive models that best adapt to the data to be analyzed are defined from the reference samples. At the Adaptive Prediction stage, the modeling strategies are calibrated aiming on improving the forecasts to be made from the next observations. Therefore, the TSFRESH features extracted from the time series to be analyzed lie the grounds of the forecast models that drive the inference of next observations. Finally, at Classification stage it is decided the significance of the registered prediction errors, hence leading to discover unexpected behaviors (discordant) that provide suspicious activity indicators. The following describes in detail each data processing stage.

9.4.1 Training and forecast method detection

The analytic dedicated server provides a battery of forecasting procedures implemented in the Section 6.3.2, which gathers among others, models based on moving averages, autoregression or smoothing. In order to adapt the detection strategy to the non-stationarity of the monitoring environment, prior to infer the traffic behavior the

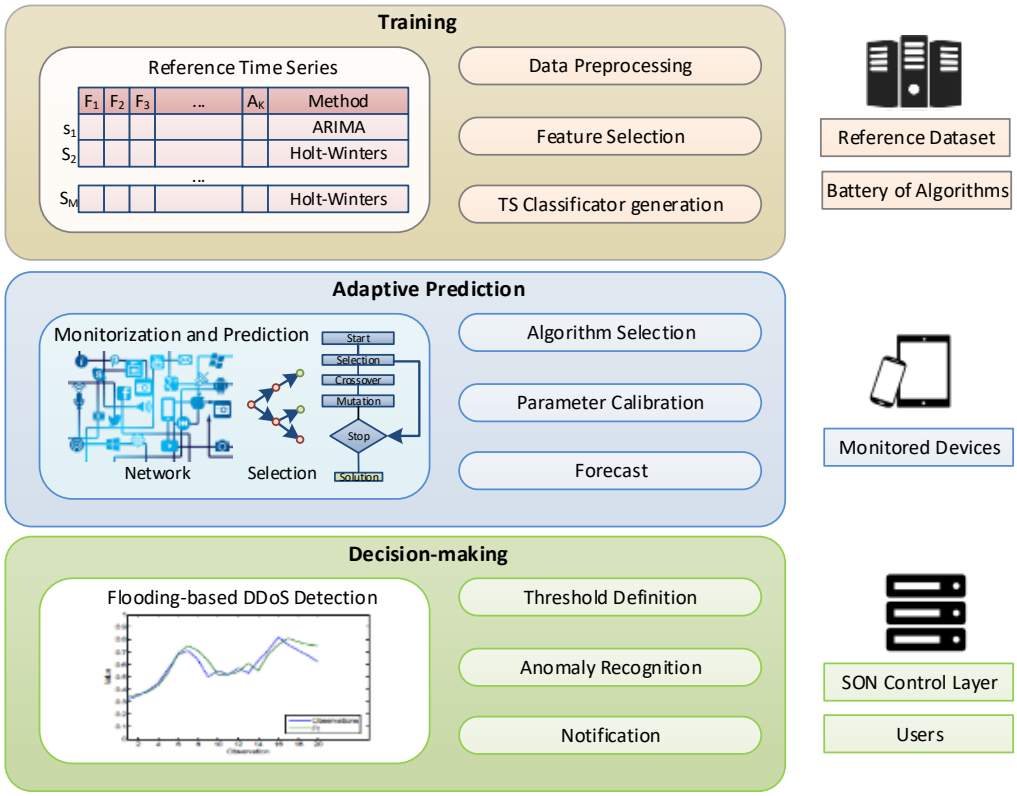


Figure 9.2: SON data processing stages.

best suited prediction method is selected and properly calibrated. Therefore, a collection of reference samples is required from which it is possible to extract the most relevant characteristics and build the classifier that stablish the best forecast function from them [TF1].

At the training stage (Figure 9.3), the classifier that decides the best prediction method is constructed. The is proposal adopts as classification procedure the Random Forest approach described by Breiman [Bre01]. According with Breiman, a random forest is a classifier consisting of a collection of tree-structured predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. In particular, the original approach implemented the variation of Classification And Regression Trees (CART) [RJPD14b] that choose which variable to split on using a greedy algorithm that minimizes error. This task typically requires specifying several adjustment parameters, for example, the maximum number of iterations to be performed as stop condition, the number of trees to construct or their maximum depth. But, as highlighted by Breiman, the number m of randomly selected attributes is the only adjustable parameter to which random forests are somewhat sensitive. This value determines the correlation between each pair of trees and the strength of each individual tree. By increasing the aforementioned parameter, both correlation and strength increase. When the correlation grows, the forest error rate increases; in the opposite, when strength grows the forest error rate decreases, so the level of both features must be balanced. This problem is addressed by applying the solution proposed by Breiman (i.e. $m = logM + 1$,

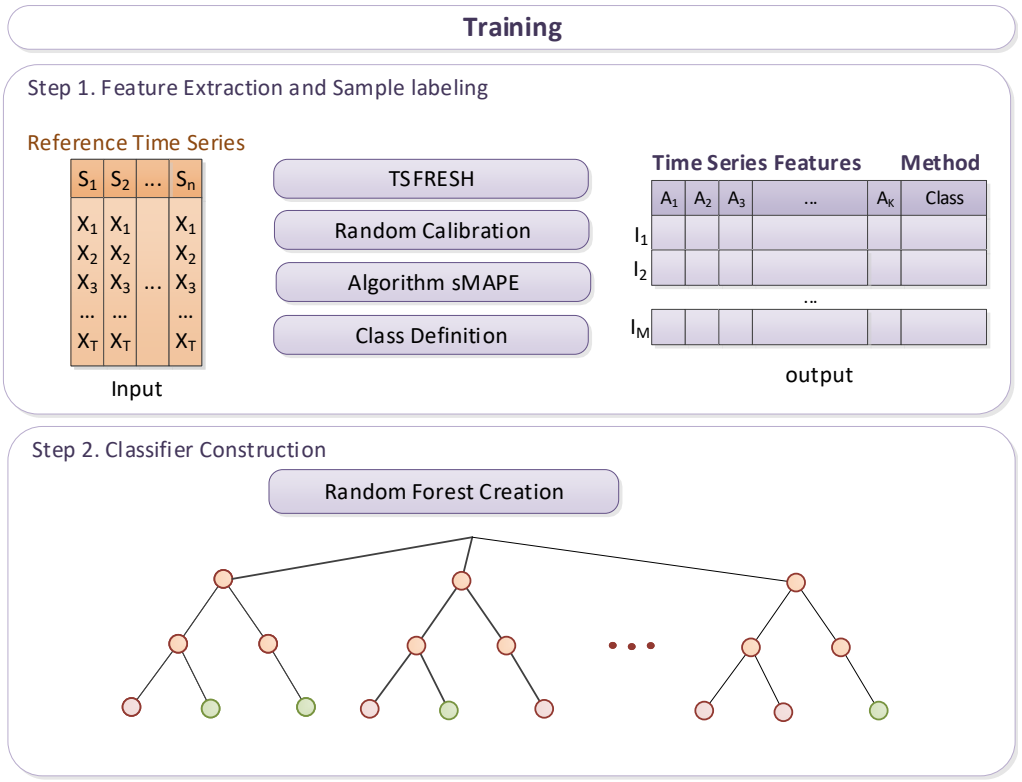


Figure 9.3: Training stage

where M is the number of features of samples within the dataset), hence postponing for future versions the implementation of alternative calibration strategies.

Each training sample considered for Random Forest definition is represented by the 100 TSFRESH attributes extracted from the reference time series (M3-Competition). The instance belongs to the class that represents the prediction algorithm that registered less significant forecasting errors when building its prediction model. The class of each sample is obtained by analyzing the time series with the complete prediction algorithm battery provided in Section 6.3.2. The solution that resulted in the lower Symmetric Mean Absolute Percentage Error (sMAPE) becames the label of the instance. Note that the sMAPE criterion was previoulsy adopted among others by the M3-Competition [Mak00], in this way enabling the evaluation of the effectiveness of different forecasting procedures. Finally, it should be remarked that one of the main disadvantages of the Random Forest classifiers is their trend towards overfitting. In this proposal, this problem is reduced by including a pre-selection step conducted by the greedy algorithm for feature discrimination described in [HLSR18] and its evaluation based on the significance of the prediction errors [Hal99].

9.4.2 Adaptive Prediction

Holte denoted in [Hol93] that pattern recognition conventionally considered that the reference datasets applied for training purposes are representative of the expected observations at the monitored environment. The presence of gradual changes over time in

the statistical characteristics of the class to which an observation belongs are the result of non-stationary fluctuations, which leads among others to the *concept drift* problem (i.e. the models built at training do no longer represent the situations that they originally intended depict). From the standpoint of the research community, the stationarity in communication networks is questionable [Mas09], which only should be assumed in very specific circumstances [MDB17]. Because of this, and in order to provide an effective defense against DDoS in any emerging scenario, non-stationarity operation is assumed. It is important to highlight that O'Reilly et al. [OGIR14] distinguished two major approaches to this problem: passive and active. The active solutions require the previous recognition of inflection points that foresee relevant changes on the monitored environment, from them updating the previously built models. Because of their modus operandi, the active solutions are usually refereed as detection and response methods. On the other hand, the passive approach assumes that the monitored feature distribution steadily varies over time, hence demanding the continuous recalibration of the analytic capabilities. Therefore, while active solutions focus on punctual drift distinction, the passive approaches proved greater effectiveness when forecasting gradual drift and recurring concepts [WIY03]. The existence of stealthy flooding-based DDoS threats based on hiding abrupt variations in the data volume injected [FR15] leads to hypothesize that with the proper data granularity, the second paradigm best suites the main objectives of the performed research, so it was implemented in this work (the development of active/hybrid solutions is postponed for future work). The detection method introduced in this section adapts to non-stationarity environments in two steps: forecasting algorithm selection and calibration (see Figure 9.4). They are described below.

9.4.2.1 Forecasting Algorithm selection

The most suitable forecast algorithm is decided based on studying the TSFRESH features extracted from the monitored time series, which serve as input of the Random Forest previously built at Training stage. The resultant class refers to the best forecast method, on which the expected behavior of the network is estimated. This procedure is repeated on each observation, so the prediction method will vary as the traffic distribution changes. For example, let the time series that represent the number of outgoing bytes (nB_{out}) of certain endpoint. The Random Forest classifier initially decided that the best suited prediction algorithm is the Simple Exponential Smoothing (SES). But at the next observations, the time series significantly gains in trend and seasonality, so the likelihood of budging from SES to Triple Exponential Smoothing (TES) increases, since TES typically behaves more accurately than SES under the new conditions [Win60].

9.4.2.2 Calibration

Most of the prediction methods in the implemented battery of algorithms required a previous configuration, where the proper calibration of its adjustment parameters plays an essential role in the achieved performance. Because of this, once the prediction method is selected, it is calibrated driven by a basic Genetic Algorithm (GA) [KRG01].

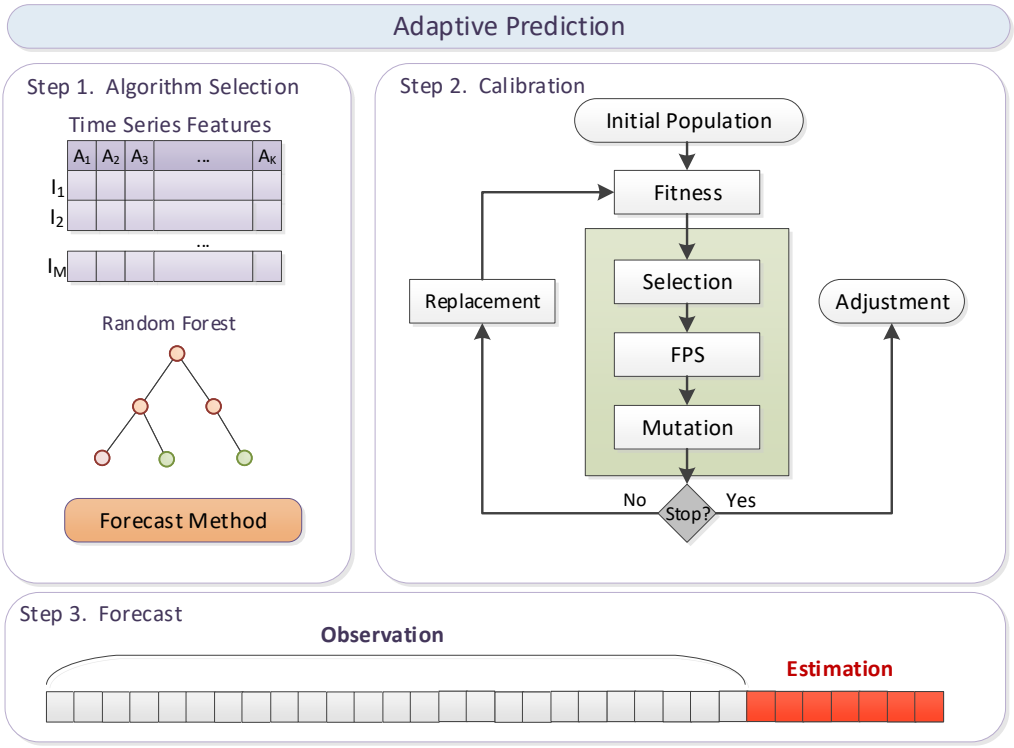


Figure 9.4: Adaptive Prediction stage

This solution is inspired by the biological evolutive theories and their genetic-molecular basis. Consequently, and in contrast to other proposals with similar purposes, the genetic algorithms are probabilistic algorithms that conduct the evolution of an initial population of individuals (observations) generated from initial factual knowledge, through actions with arbitrary results (i.e. genetic mutations and gen recombination) that try to get closer to the optimal solution in each iteration, hence resembling those of the biological evolution processes. Its main drawbacks are related with high resource consumption and not guarantee of finding and optimal solution, both of them extensively discussed in the bibliography [ESE14]. But their discussion and mitigation are out of the scope of the principal contributions of this research.

The proposed detection approach implements a GA as solution to the forecasting algorithm calibration problem after taking into consideration different reasons, highlighting among them: the fact that GAs already posed solutions to optimization problems previously proved with calibration purposes [Rui16], they are capable of operate on vectors of adjustment parameters of different nature, and their operation adapts to the detail level in which calibrations must be calculated. Note that the latter is especially valuable when operating in real time scenarios, hence allowing to balance accuracy and performance to satisfy the implemented security management policies.

In the implemented GA, it is considered as evolving population the set of candidate adjustments, where each individual raises a possible solution. Their genotype represents a vector of gens in which each position contains one of the adjustment parameters of the prediction method. For example, in the case of TES a collection of four characteristics

Table 9.2: Summary of the GA calibration

Feature	Highlights
Individual	An individual represents a possible calibration. The genotype is a vector where each gen is an adjustment parameter of the prediction algorithm.
Initial Population	At the first launch, the initial population is generated by assigning randomly values to gens. The initial population of the next observation is the final population of the previous execution.
Fitness	The sMAPE [Mak00] obtained by a specific calibration.
Selection	Fitness Proportionate Selection (FPS) [GD91]
Crossover	Swapping gens from an arbitrary pivot.
Mutation	Uniform mutation of an arbitrary gen.
Stop Condition	Reached a previously defined maximum number of iterations or discovery of an optimal solution.

would be constituted: data smoothing factor (α), trend smoothing factor (β), seasonal change smoothing factor (γ) and forecast horizon (τ) [Win60]. The initial population is randomly calculated and only the most adapted individuals hold possibilities of persisting at future generations. Note that as in nature, the fitness of an individual ponders its ability to adapt to the environment, and therefore the probability of procreation. Therefore, the fitness function of the implemented GA returns the sMAPE calculated when the prediction algorithm is calibrated according to the genotype of an individual.

In addition, the GA performs simple crossover and uniform mutation per iteration. The first of them selects a couple of parents per crossover by Fitness Proportionate Selection (FPS) [GD91], then randomly deciding a swapping point and exchanging their genetical contents pivoting on such point. Consequently, the descendant individual replaces the parent with lower fitness. At the mutation stage, an arbitrary gen of the descendant is replaced by a random value. Because of the gens may present different nature, this action is constrained by the boundaries stablished by the data range of the adjustment parameter. For example, the α parameter of the TES prediction function ranges in $0 \ldots 1$, so the random uniform mutations on alpha must be restricted to $0 \ldots 1$. The algorithm has two stop conditions: a predefined maximum number of iterations (worst case), and some individual reaching its optimal fitness, i.e. sMAPE=0.

From the prediction algorithm selected by the Random Forest classifier, as well as from its calibration according to the adjustment indicated by the best individual of the final population, the next h observations of the time series to be analyzed are estimated. The final population is temporally stored for serving as initial population for the next execution of the GA, in this way usually gaining accuracy. Note that this decision is based on the fact that most of the time series will be similar, so large changes in the adjustment values are not expected. Table 9.2 summarizes the main steps of the implemented GA.

9.4.3 Classification

At the classification stage, the natures of the time series of aggregated metrics constructed from the monitored traffic flows are decided. In this context, it is assumed that an observation is an outlier if it matches with an unexpected behavior, i.e. when the variation

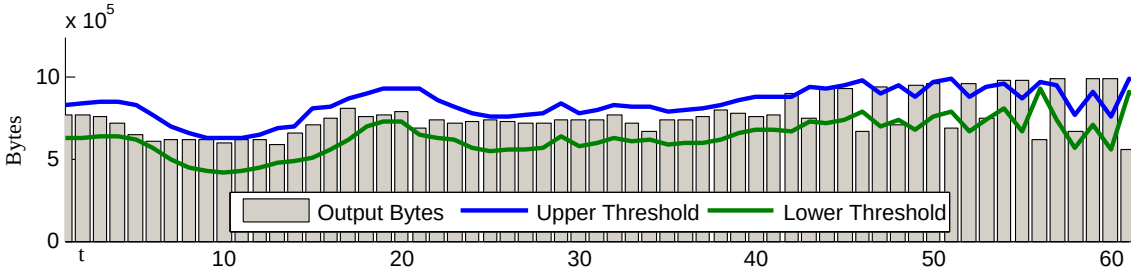


Figure 9.5: Example of outlier identification.

between a prognosis at certain time horizon and the observed value differ significantly. Because the projection of continuous values on time tends to yield errors, the main challenge of this process is to define their relevance, which is addressed by defining adaptive thresholds. In the aftermath, outliers are tagged as potential *malicious* behaviors, and normal situations are classified as *legitimate*, so the current implementation acts as a binary classifier. The reference reasoning framework [SMMVGV17d] provides advanced analytical capabilities related with building prediction intervals, most of them widely accepted by the research community for network traffic study. Of them, it is integrated the adaptive thresholding methodology described in [MWH97], which defines the following adaptive thresholds:

$$Ath = \hat{x}_{n+1} \pm K\sqrt{\sigma^2(E_t)} \tag{9.3}$$

where $\hat{x}_{n+1}$ is the forecast of certain aggregated metric x at $n+1$ horizon, E_t is the Euclidean distance between $\hat{x}_{n+1}$ and x_{n+1}, and K is the adjustment parameter that configures the restrictiveness of the sensor. The equations distinguish an upper threshold Ath_{up} and a lower threshold Ath_{low}, both adapted to t. It is expected that the greater values of K, the higher noise tolerance, since this situation expands the margin of error between $\hat{x}_{n+1}$ and x_{n+1}. Figure 9.5 illustrates an example of outlier induced by a DDoS flooding-based attack, where at T=41 a compromised endpoint injects a large number of HTTP requests. During the attack the threshold was exceeded, which leads to label the traffic as *malicious*.

9.5 Evaluation Methodology

The collection of samples gathered for evaluation includes outgoing traffic captures from 62 different devices. Each sample was created from traffic monitorizations separated in time periods of 1, 3 and 5 days, comprising a total amount of 50 instances with 3 hours per device, so the dataset contains 3,100 samples of normal traffic (Table 9.3). At the end of each normal traffic capture, the tools described in [SLo][WcD] launched DDoS attacks; in particular, traffic injections based on UDP, HTTP or TCP flood with low, medium and high intensity. Accordingly, the dataset provides 27,900 samples with malicious contents, 9,300 per intensity (see Table 9.4). Usage profiles comprised daily user habits (i.e. general-purpose usage), synthetic web navigation with various automatization

Table 9.3: Monitored activities and endpoint devices at the performed experimentation.

No.	No.	Samples	p-ADF
Daily user habits	20	1,000	0.103
Browser bot A	4	200	0.065
Browser bot B	16	800	0.130
Browser bot C	4	200	0.008
Audio streaming	4	200	0.040
Video streaming	14	700	0.065

Table 9.4: Normal and attack traffic samples per monitored device.

Endpoint	No.	Normal	Attack
Desktop computer	29	1,450	13,050
Notebook	16	800	7,200
Smartphone	8	400	3,600
Tablet computer	6	300	2,700
Smartwatch	2	100	900
Smart TV	1	50	450

tools and multimedia streaming (audio and video). The synthetic web browsing profiles were separated according to the tool that the endpoint executed; being referred as A for *Internet Noise* [SIN], B for *Noiszy* [NZ1] and C for *TrackMeNot* [TnS]. Six families of devices were considered: desktop computers, notebooks, smartphones, tablet computers, smartwatches and smart TVs. Note that given that in terms of traffic modeling, the type of the endpoint had less impact than its usage mode, the conducted study primarily focused on their behavior. Table 9.3 displays the average p-value of the Augmented Dickey-Fuller test (ADF) that assess the no-stationarity of each traffic profile [CL95]. The p-values lower than 0.05 resemble stationary processes, which leads us to assume that most of the endpoints behave as non-stationary data sources.

The effectiveness of the proposed method has been tested by adopting an experimental evaluation methodology, in which the impact on effectiveness was measured when varying the following adjustment parameters: granularity, traffic profile and attack. In analogy with previous publications, this task was addressed by considering a classical binary classificator based on observing its sensitivity and specificity. The first of them determines the ability to properly point out anomalies as *malicious*. On the other hand, the specificity measures the ability of recognizing normal activities as *legitimate*. From their representation in the Receiver Operating Characteristic (ROC) space, several effectiveness indicators have been extracted, standing out for relevance the Area Under the Curve (AUC), and the True Positive Rates (TPR) and False Positive Rates (FPR) achieved from the best sensor adjustment in terms of K. The optimal setting coincides with the position of the ROC curve that displays the better Youden index [BNR14] (Y), which ranged from -1 (worst) to 1 (optimal).

Table 9.5: AUC registered per observation granularity when varying K.

Indicator	Observation granularity					
	7.5 Sec.	15 Sec.	30 Sec.	1 Min.	2 Min.	3 Min.
nP_{in}	0.75	0.79	0.72	0.75	0.69	0.69
nP_{out}	0.84	0.95	0.93	0.93	0.83	0.80
nB_{in}	0.61	0.65	0.63	0.67	0.65	0.62
nB_{out}	0.75	0.91	0.87	0.84	0.76	0.72
$H(nP_{in})$	0.63	0.71	0.75	0.67	0.69	0.67
$H(nP_{out})$	0.65	0.74	0.72	0.69	0.68	0.66
$H(nB_{in})$	0.64	0.71	0.75	0.65	0.69	0.65
$H(nB_{out})$	0.63	0.73	0.72	0.71	0.69	0.66
nMSE(nP)	**0.88**	**0.96**	**0.94**	**0.95**	**0.85**	**0.89**
$nMSE(nB)$	0.60	0.68	0.68	0.74	0.68	0.61

9.6 Results

The obtained results are summarized in the following subsections.

9.6.1 Impact of granularity

At this experiment, the accuracy of the sensor was measured when studying traffic flows captured in time intervals of 7.5 sec., 15 sec., 30 sec., 1 min., 2 min. and 3 min; hence only focusing on the monitorization interval and the adjustment of the K parameter for adaptive thresholding calibration. The accuracy achieved per metric and configuration is illustrated in Table 9.5, where the effectiveness of this proposal is expressed in terms of AUC, and the corresponding TPR and FPR for the best granularity is shown in Table 9.6. This performance indicator was calculated via trapezoidal approximation with 0.005 estimated error. The best studied granularity was 15 seconds per observation, which provides the most accurate results (AUC=0.96, TPR=0.93 and FPR=0.01). When the granularity is lower (i.e. the duration of the observation is smaller), the accuracy worsens. For example, when 7.5 seconds the best registered AUC was 0.88. Similarly, as the level of detail falls, the effectiveness of the proposal decreases, hence reaching AUC=89.2 when 3 minutes per observations. This is due to the fact that with small observations the information they compile tends to be less significant, hence being more likely to infer noise. On the contrary, when the observation is too large, the first observations of the attack may go unnoticed among legitimate traffic. In this case, the adaptation to non-stationarity readjusted the analytic algorithms, so if the attack is not initially detected, it may be considered part of the normal activity of the network. Finally, it is worth to highlight the accuracy achieved by metrics directly related to the total incoming (nP_{in}) and outcoming (nP_{out}) packet, which divergence ($nMSE(nP)$) behaved as the most accurate DDoS indicator at the performed experimentation. In contrast with the classical entropy-based DDoS detection solutions focused on intermediate/victim edge audition, these metrics proved not to be as effective at single source-side monitorization.

Table 9.6: Best parameters

Best(15 sec.)		
TPR	FPR	Y
0.56	0.01	0.55
0.92	0.01	0.91
0.74	0.51	0.23
0.83	0.01	0.82
0.81	0.48	0.33
0.87	0.49	0.38
0.80	0.47	0.33
0.85	0.49	0.36
0.93	**0.01**	**0.92**
0.76	0.50	0.26

Table 9.7: AUC registered per traffic profile at 15 sec. granularity.

Indicator	Traffic usage profile					
	P_0	P_1	P_2	P_3	P_4	P_5
nP_{in}	0.86	0.84	0.86	0.94	0.70	0.84
nP_{out}	0.96	0.96	0.96	0.97	0.94	0.96
nB_{in}	0.79	0.71	0.51	0.93	0.81	0.67
nB_{out}	0.92	0.88	0.95	0.93	0.92	0.93
$H(nP_{in})$	0.73	0.54	0.80	0.84	0.57	0.67
$H(nP_{out})$	0.73	0.76	0.79	0.72	0.63	0.67
$H(nB_{in})$	0.70	0.58	0.79	0.83	0.56	0.67
$H(nB_{out})$	0.71	0.78	0.80	0.64	0.64	0.66
nMSE(nP)	**0.96**	**0.97**	**0.97**	**0.97**	**0.96**	**0.96**
$nMSE(nB)$	0.84	0.76	0.36	0.91	0.86	0.72

9.6.2 Impact of traffic profile

With the purpose of facilitating the understanding of the achieved results, the impact of the device usage mode on the effectiveness has been studied when assuming the best granularity of the previous experimentation, as it is illustrated in Table 9.7. The six traffic activity profiles described in Table 9.3 were analyzed, hence leading to the following best results: user daily habits P_0 (AUC=0.96), A synthetic traffic P_1 (AUC=0.97), B synthetic traffic P_2 (AUC= 0.97), C synthetic traffic P_3 (AUC=0.97), audio streaming P_4 (AUC=0.96) and video streaming P_5 (AUC=0.96). Note that similarly to the previous tests, the best metric is often the difference between incoming and outgoing packets ($nMSE(nP)$), which is closely followed by the total number of incoming packets (nP_{in}) and the total number of outgoing packages (nP_{out}). Again entropy-based metrics have not been effective enough. Since no significant variations have been recorded between traffic profiles, it is possible to conclude that the proposed detection method was capable of self-calibrating according to the traffic distribution inherent to each type of endpoint, in this way posing an effective solution regardless the nature of the device.

Table 9.8: AUC registered per attack type at 15 sec. granularity.

Indicator	Attack type								
	H_l	H_m	H_h	T_l	T_m	T_h	U_l	U_m	U_h
nP_{in}	0.90	0.87	0.83	0.80	0.94	0.89	0.95	0.76	0.75
nP_{out}	1.00	1.00	1.00	0.86	1.00	1.00	1.00	0.98	0.99
nB_{in}	0.87	0.76	0.73	0.62	0.72	0.61	0.69	0.61	0.61
nB_{out}	1.00	0.99	0.99	0.81	0.97	0.99	0.96	0.96	0.97
$H(nP_{in})$	0.65	0.68	0.78	0.75	0.80	0.79	0.61	0.67	0.74
$H(nP_{out})$	0.42	0.69	0.92	0.71	0.64	0.79	0.55	0.70	0.84
$H(nB_{in})$	0.66	0.65	0.77	0.73	0.83	0.78	0.58	0.65	0.73
$H(nB_{out})$	0.44	0.69	0.92	0.73	0.64	0.71	0.57	0.72	0.86
nMSE(nP)	**0.99**	**1.00**	**1.00**	**0.89**	**0.99**	**1.00**	**1.00**	**1.00**	**1.00**
$nMSE(nB)$	0.79	0.78	0.68	0.66	0.56	0.70	0.80	0.71	0.72

9.6.3 Impact of attack type

Table 9.8 summarizes the accuracy obtained by threat group, which include flooding-based low-rate attacks on HTTP (H), TCP (T), UDP (U) protocols. They have been clustered based on intensity, hence distinguishing three subsets: high intensity (h), medium (m) and low (l). For example, the symbol T_l refers to the group of TCP attacks of low intensity. In general terms, the effectiveness was better than at previous tests, where the metrics $nMSE(nP)$, (nP_{in}) and (nP_{out}) outstand. The best AUC ranged from 0.99 to 1.0 regardless the intrusion subset. This obvious improvement is empowered by a fundamental characteristic of the test: the K adjustment factor that was applied for configuring its restriction level now is set to detect a specific menace; this did not happen at the second experiment, where the same threshold distance was configured for all the DDoS methods. In view of these results, it is possible to deduce that the proposed method has been able to adapt to each attack group. However, as the threat specificity decreases it is tended to lose precision. This must be taken into account when proposing general-purpose self-organizing defenses, where it might be advisable to deal with the different intrusion categories separately.

9.7 Final Remarks

This chapter introduced an autonomic architecture with operability on the emerging communication networks and a novel intrusion detection approach adaptable to non-stationary processes. It has posed the most innovative aspects on the analysis of DDoS attacks on the source side, hence inferring whether a network endpoint is participating in a DDoS attack. The proposal effectiveness has been proven by an extensive experimentation, where traffic from 62 devices of different nature has been monitored and analyzed looking for DDoS traits. The obtained results have demonstrated that it is possible to lay detection strategies based on predictive analysis on the basis of different flow-level metrics measured in the network. Nevertheless, not all the metrics have shown equal effectiveness. For

example, those based on studying the proportionality between incoming and outgoing traffic yielded promising results, from which it follows that solutions like could be successfully accommodated for operating on heterogeneous and non-stationary network environments. In contrast, the classical entropy-based approaches for DDoS detection not resulted as effective. This fact brings uncertainty about their accuracy when acting at source-side observations, more particularly at non-stationary contexts. The performed research on this subject has thereby introduced an innovative approach compared to other proposals, while proving promising results in the meantime.

Chapter 10

Conclusions and Future Work

The fifth generation of mobile networks has emerged pushed by emerging communication contexts, where ambitious performance indicators should be addressed to ensure ubiquitous access to network services. 5G networks are in consequence foreseen as the technological platform to sustain the economic and societal challenges raised by the modern communication landscape. Bearing this context in mind, the presented research has been framed into the study of knowledge acquisition processes aimed to provide analytical capabilities for conducting a self-organizing management approach in 5G networks.

5G supportive technologies such as SDN, NFV, SON, cloud computing and machine learning have provided the architectural principles for designing and implementing the analytical methods introduced throughout this thesis. To accomplish the use cases goals, the 5G reference architecture provided by the SELFNET project has been considered. There, particular interest has been set on the Analysis component, where the situational awareness reasoning has led to conduct strategies for detecting discordant behaviors in the monitored networks, thus laying the proposal of a reasoning and knowledge acquisition framework for 5G analytics. On the other hand, the detection of network threats driven by autonomic incident management approaches have been analyzed in the proposals oriented to deal with Economic Denial of Sustainability (EDoS) and Distributed Denial of Service (DDoS). All of them have been widely discussed and effectively assessed throughout this research.

To deal with the challenge of performing advanced network analysis, a reasoning and knowledge acquisition framework has been introduced. It has been adapted to support analytic methods on 5G monitoring environments toward the provision of self-organizing autonomic capabilities. Because of this, the framework assumes the design principles of the new generation networks and the technologies they implement. This fact motivated its instantiation tightly coupled to the SELFNET architecture, which involved the implementation of different components based on pattern recognition methods, prediction algorithms, adaptive thresholding approximations and knowledge inference techniques. Such components were orchestrated as a modular architecture, easily expandable and scalable, adapted to the large volume of information flowing through the network. A detailed evaluation of those capabilities has been carried out to validate the effectiveness of the proposal; both at component level, being assessed against reference datasets, and

at use case level, by analyzing behavioral aspects of the network traffic.

Furthermore, an entropy-based model for the detection of EDoS attacks in cloud environments has been introduced in this thesis. For this purpose, a comprehensive revision of the EDoS related research has been covered to elaborate a multi-layered architecture tackling the detection of EDoS attacks. The proposed work suggested good detection accuracy, thus preventing the unnecessary consuption of additional cloud instances if they were issued by auto-scaling policies based on unreal demands. The experiments conducted to validate the proposed architecture have encompassed all the stages defined in the architecture, starting from the monitoring and aggregation of metrics that directly affect the customer's cost model, implementing the novelty detection procedures to recognize an EDoS attack, and enabling decision-making and notification actions to be applied in the system. It is also worth mentioning the distinctive approach of the proposed model compared to other resource-consuming methods presented in the literature such as those based on analyzing requests of large files, triggering costly database queries, exploiting web vulnerabilities, etc. Such detection enhancement is achieved since this architecture relies on server-side consumption analysis rather than anomalous network-level metric patterns.

Moving a step further in the analysis of EDoS threats, they have been studied in the self-organizing network context by highlighting its adaptation to 5G network scenarios. This study has led to identify two main categories of EDoS threats. The first of them is W-EDoS, which aims on exploiting vulnerabilities by workload injection in NFV auto-scaling policies. On the other hand, the I-EDoS threats take advantage of the NFV auto-instantiation capabilities by thwarting the orchestration processes. Consequently, a pair of strategies for their mitigation have been proposed. They were based on modeling the normal behavior of the protected system and discovering discordant activities. Their effectiveness have been proven at the performed preliminary experimentation, which considered different adjustment parameters; among them, the attack intensity, the normal traffic features, and a confidence interval for adaptive thresholding and entropy degree.

Finally, the DDoS use case has been studied by the proposal of a detection approach for inferring the participation of network endpoint in DDoS attacks. It raised a flooding-based detection approach of DDoS attacks by analyzing source-side traffic flows from protected devices, in this way supporting the development of defensive self-organized solutions grounded on endpoint monitorization. To this end, the detection scheme implemented predictive analysis considering the non-stationarity of the analyzed traffic, where adaptive prediction has been implemented by examining time series features that allowed the selection of the most suited prediction algorithm, and its calibration conducted by a genetic algorithm. Several flow-level metrics have been analyzed and studied to infer the participation of a protected device in an orchestrated DDoS attack. The obtained results demonstrated high accuracy, hence demonstrating that it is possible to raise similar solutions to the challenges inherent to this type of network threats. Another interesting finding is that the proposal behaved almost indistinctly on different network usage profiles, whether they pose legitimate or malicious activities.

In the light of the results obtained by the performed research, it is possible to conclude

that advanced analytical capabilities in emerging network contexts firmly contribute to achieve the challenging requirements of network autonomic management introduced at the begining of this thesis.

Last, but not least, it is also important to bear in mind that 5G is still in research stage. Thus, various architectural elements of the 5G architecture are foreseen to be integrated at industry-level by the year 2020 to accomplish the goals of this emerging platform. Such integration is nowadays being prototyped by the researchers in compliance with the requirements of 5G networks, as it was explained throughout this thesis. The final goal is the exploration of innovative approaches for extending the current capabilities of 4G/LTE architectures towards the disruptive technological that 5G supposes. Even though a primary objective is to perform concept-proof validations, larger industry-level scenarios have been studied with the use of well-known datasets such as CAIDA, NSL-KDD or M3 competition. Those are widely accepted by the research community, and provide real data on which the experimental results are validated with more confidence. On the other hand, the ability to generate large data samples with adjustment to real experimental scenarios allow a better validation of the proposals. That is the case of the EDoS detection proposals where data heterogeneity and statistical distribution of the sampled observations have been addressed, thus dismissing the need of external datasets. Because of the experimental rigor, the proposals prevent the reader their generalization or biased interpretation by delimiting their scope. In the same way, the assumptions, limitations and design principles have been explicitly introduced on each for providing the operational landscape where the contributions of this thesis have proven their validity.

10.1 Future Work

Bearing in mind the novelty of the introduced proposals and the level of maturity of their supportive technologies, promising research subjects have been raised as a result of the conducted research. The knowledge acquisition approach aimed on the development of 5G network analytics have exposed new research lines. The clearest of them is to delve into how the knowledge acquisition framework can be instantiated in order to face the challenges posed by the different use cases. These may have very different requirements; for example, a use case focused on bandwidth optimization may require information that facilitates deciding proactive actions, and therefore must be primarily based on prediction; but the reactive responses have greater impact on mitigating threats such as botnets or DoS attacks. Other interesting topics, such as inclusion of data protection policies or the communication ways between the proposed framework and the rest of network components (protocols, interfaces, etc.), have not been detailed throughout the article, in this way postponing their development for future work.

Regarding the detection of EDoS threats, the presented approach, evaluation methodology and the conducted experimentation posed also new potential research lines. On the one hand, experimental scenarios should be extended to couple diverse network conditions to either enhance the validation or to disclose some evasion techniques. Moreover, the defined model of measuring the resource consumption and diagnosing its

entropy can be accommodated to include additional metrics, thus extending its scope to wider analysis scenarios. Furthermore, the proposed method might be fitted to enhance adaptive auto-scaling policies on cloud platforms by incorporating more complex evaluation criteria. Complementary, the existing decision-making and countermeasures to EDoS attacks remain far from being evolved, and might effectively complement the conducted research.

It was also identified when analyzing the impact of EDoS threats in self-organized networks that in certain circumstances the proposal reported significant false positive rates, which raised interesting lines of future work. For example, integration of alternative analytic techniques, granularity optimization or adaptation to non-stationary processes. In addition, for the better understanding of our contributions several issues have not been addressed, which were outlined throughout the document. They included among others robustness against adversarial threats or the assumption of data protection policies, hence being relegated to forthcoming research.

In addition, one of the main drawbacks noticed when assessing the DDoS detection proposal was its proven tendency to loss in precision as its specificity grows, i.e. when it is trained to act against a larger variety of attacks. This feature raises an interesting line of future research that leads to encourage outlining ensemble learning methods for providing a general-purpose defensive solution. Throughout the research alternative ways of improvement have been highlighted, being of special interest those based on expanding the diversity of metrics, prediction algorithms and adjustment parameters. It is expected that as a result of these enhancements, a greater effectiveness may be registered. This also contributes to the clearer understanding of the studied traffic profiles, as well as to assess their impact on intrusion detection at forthcoming communication networks.

Finally, there are common design and implementation aspects that might enhance the overall performance and accuracy throughout the proposals presented so far. Such is the case of secure software development which should guarantee data integrity when performing analytical tasks. For instance, handling efficient data structures suited for supporting heterogenous data types should raise the resulting performance of the proposed solutions. Likewise, the use of authenticated software interfaces will raise the restrictiviness level that prevent third parties for accessing or manipultaing the acquired data. Thereby, it should allow a more trusted execution environment on which knowledge generation takes place. Those aspects represent interesting lines of work that might be addressed on future implementations.

Bibliography

[5G-16a] 5G-PPP. 5G PPP use cases and performance evaluation models. https://5g-ppp.eu/wp-content/uploads/2014/02/5G-PPP-use-cases-and-performance-evaluation-modeling_v1.0.pdf, April 2016.

[5G-16b] 5G-PPP. 5G-PPP: View on 5G Architecture. https://5g-ppp.eu/wp-content/uploads/2014/02/5G-PPP-5G-Architecture-WP-For-public-consultation.pdf, July 2016.

[5G-18] 5G-PPP. Advanced 5G Network Infrastructure for the Future Internet. https://5g-ppp.eu/wp-content/uploads/2014/02/Advanced-5G-Network-Infrastructure-PPP-in-H2020_Final_November-2013.pdf, January 2018.

[AAA14] A. A Atayero, O. I. Adu, and A. A. Alatishe. Self organizing networks for 3GPP LTE. In *Proceedings of the International Conference on Computational Science and Its Applications*, pages 242–254, Guimaraes, Portugal, June 2014.

[AAB13] Wael Alosaimi and Khalid Al-Begain. A new method to mitigate the impacts of the economical denial of sustainability attacks against the cloud. In *Proceedings of the 14th Annual Post Graduates Symposium on the convergence of Telecommunication, Networking and Broadcasting (PGNet)*, pages 116–121, Liverpool, UK, June 2013.

[AAB+17] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, Luca Invernizzi, Michalis Kallitsis, et al. Understanding the mirai botnet. In *Proceedings of the 26th USENIX Security Symposium*, pages 1092–1110, Vancouver, BC, Canada, August 2017.

[ABC+14] J. G. Andrews, S. Buzzi, W. Choi, S. V Hanly, A. Lozano, A. CK Soong, and J. C. Zhang. What will 5G be? *IEEE Journal on selected areas in communications*, 32(6):1065–1082, June 2014.

[Acw] The Attribute-Relation File Format (ARFF). http://www.cs.waikato.ac.nz/ml/weka/arff.html.

[ADAH14] Basheer N Al-Duwairi and Ahmad T Al-Hammouri. Fast flux watch: a mechanism for online detection of fast flux networks. *Journal of advanced research*, 5(4):473–479, July 2014.

[Afe16] Afek, Yehuda and Bremler-Barr, Anat and Harchol, Yotam and Hay, David and Koral, Yaron. Making DPI Engines Resilient to Algorithmic Complexity Attacks. *IEEE/ACM Transactions on Networking*, 24(6):3262–3275, December 2016.

[AHSS12] Fahd Al-Haidari, Mohammed H Sqalli, and Khaled Salah. Enhanced edos-shield for mitigating edos attacks originating from spoofed ip addresses. In *Proceedings of the IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 1167–1174, Liverpool, UK, June 2012.

[AIIE13] O. G. Aliu, A. Imran, M. A. Imran, and B. Evans. A survey of self organisation in future cellular networks. *IEEE Communications Surveys & Tutorials*, 15(1):336–361, 2013.

[Aka71] Hirotugu Akaike. Autoregressive model fitting for control. *Annals of the Institute of Statistical Mathematics*, 23(2):163–180, December 1971.

[Akf] Apache Kafka: A distributed streaming platform. `https://kafka.apache.org`.

[ALW+14] Ian F Akyildiz, Ahyoung Lee, Pu Wang, Min Luo, and Wu Chou. A roadmap for traffic engineering in sdn-openflow networks. *Computer Networks*, 71:1–30, October 2014.

[AmW] Amazon Web Services. `https://aws.amazon.com`.

[AR14] Mohammed N Alenezi and Martin J Reed. Uniform dos traceback. *Computers & Security*, 45:17–26, September 2014.

[ArP] ARCADIA: A Novel Reconfigurable by Design highly Distributed Applications Development Paradigm Over Programmable Infraestructure, Funded under: H2020-ICT-2014-1. Project Reference 645372. `http://www.arcadia-framework.eu`.

[ARS16] Mamta Agiwal, Abhishek Roy, and Navrati Saxena. Next generation 5g wireless networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 18(3):1617–1655, February 2016.

[ASMW15] Aya A Aly, Nesma A Saleh, Mahmoud A Mahmoud, and William H Woodall. A reevaluation of the adaptive exponentially weighted moving average control chart when parameters are estimated. *Quality and Reliability Engineering International*, 31(8):1611–1622, December 2015.

[AWH+17] Rana Aamir Raza Ashfaq, Xi-Zhao Wang, Joshua Zhexue Huang, Haider Abbas, and Yu-Lin He. Fuzziness based semi-supervised learning approach for intrusion detection system. *Information Sciences*, 378:484–497, February 2017.

[AY01] Charu C Aggarwal and Philip S Yu. Outlier detection for high dimensional data. In *Proceedings of the ACM SIGMOD international conference on Management of data (SIGMOD'01)*, volume 30, pages 37–46, Santa Barbara, CA, USA, May 2001.

[AYON17] Wathiq Laftah Al-Yaseen, Zulaiha Ali Othman, and Mohd Zakree Ahmad Nazri. Real-time multi-agent system for an adaptive intrusion detection system. *Pattern Recognition Letters*, 85:56–64, January 2017.

[BAG15] Noam Ben-Asher and Cleotilde Gonzalez. Effects of cyber security knowledge on attack detection. *Computers in Human Behavior*, 48:51–61, July 2015.

[Bar17] Barona López, L. I. and Maestre Vidal, J. and García Villalba, L. J. An Approach to Data Analysis in 5G Networks. *Entropy*, 19(2):74, February 2017.

[BBBS17] Anat Bremler-Barr, Eli Brosh, and Mor Sides. Ddos attack on cloud auto-scaling mechanisms. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM 2017)*, pages 1–9, Atlanta, GA, US, May 2017.

[BBK14] Monowar H Bhuyan, Dhruba Kumar Bhattacharyya, and Jugal K Kalita. Network anomaly detection: methods, systems and tools. *IEEE communications surveys & tutorials*, 16(1):303–336, First Quarter 2014.

[BBK15] Monowar H Bhuyan, DK Bhattacharyya, and Jugal K Kalita. An empirical evaluation of information metrics for low-rate and high-rate ddos attack detection. *Pattern Recognition Letters*, 51:1–7, January 2015.

[BCH+17] Nicola Bui, Matteo Cesana, S Amir Hosseini, Qi Liao, Ilaria Malanchini, and Joerg Widmer. A survey of anticipatory mobile networking: Context-based classification, prediction methodologies, and optimization techniques. *IEEE Communications Surveys & Tutorials*, 19(3):1790–1821, April 2017.

[BF85] Leo Breiman and Jerome H Friedman. Estimating optimal transformations for multiple regression and correlation. *Journal of the American statistical Association*, 80(391):580–598, August 1985.

[BG16] Anna L Buczak and Erhan Guven. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials*, 18(2):1153–1176, October 2016.

[BKH+14] Arsany Basta, Wolfgang Kellerer, Marco Hoffmann, Hans Jochen Morper, and Klaus Hoffmann. Applying NFV and SDN to LTE mobile core gateways, the functions placement problem. In *Proceedings of the 4th workshop on All things cellular: operations, applications, & challenges*, pages 33–38, Chicago, Illinois, USA, August 2014.

[BL02] Hyeran Byun and Seong-Whan Lee. Applications of support vector machines for pattern recognition: A survey. *Pattern Recognition with Support Vector Machines, Lecture Notes in Computer Science LNCS*, 2388:213–236, July 2002.

[BLVCMV+17] Lorena Isabel Barona Lopez, Angel Leonardo Valdivieso Caraguay, Jorge Maestre Vidal, Marco Antonio Sotelo Monge, and Luis Javier García Villalba. Towards incidence management in 5G based on situational awareness. *Future Internet*, 9(1):3, January 2017.

[BLVCSMGV16] L. I. Barona López, A. L. Valdivieso Caraguay, M. C. Sotelo Monge, and L. J. García Villalba. Key Technologies in the Context of Future Networks: Operational and Management Requirements. *Future Internet*, 9(1):1–15, October 2016.

[BM15] Parminder Singh Bawa and Selvakumar Manickam. Critical Review of Economical Denial of Sustainability (EDoS) Mitigation Techniques. *Journal of Computer Science*, 11(7):855–862, November 2015.

[BNR14] Leonidas E Bantis, Christos T Nakas, and Benjamin Reiser. Construction of confidence regions in the ROC space after the estimation of the optimal Youden index-based cut-off point. *Biometrics*, 70(1):212–223, March 2014.

[Bre96] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, August 1996.

[Bre01] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, October 2001.

[Bro57] Robert G. Brown. Exponential smoothing for predicting demand. In *Operations Research*, volume 5, pages 145–145, 1957.

[BS15] A. Sukhada Bhingarkar and B. Deven Shah. A survey: Securing cloud infrastructure against edos attack. In *Proceedings of the International Conference on Grid Computing and Applications (GCA)*, pages 16–22, Athens, Greece, January 2015.

[BSB16] Zubair A Baig, Sadiq M Sait, and Farid Binbeshr. Controlled access to cloud resources for mitigating Economic Denial of Sustainability (EDoS) attacks. *Computer Networks*, 97:31–47, March 2016.

[BSMT14] Sajal Bhatia, Desmond Schmidt, George Mohay, and Alan Tickle. A framework for generating realistic traffic for Distributed Denial-of-Service attacks and Flash Events. *Computers & Security*, 40:95–107, February 2014.

[BTAS14] B. Bangerter, S. Talwar, R. Arefi, and K. Stewart. Networks and devices for the 5G era. *IEEE Communications Magazine*, 52(2):90–96, February 2014.

[BTI+03] Chadi Barakat, Patrick Thiran, Gianluca Iannaconne, Christophe Diot, and Philippe Owerzazki. Modeling internet backbone traffic at the flow level. *IEEE Trans. on Signal Processing*, 51(8):2111–2124, 2003.

[BWSMea17] P. Bisson, J. Wary, M. A. Sotelo Monge, and et. al. 5G PPP Phase 1 Security Landscape. White Paper, 5G PPP Security Work Group, June 2017.

[CB10] NM Mosharaf Kabir Chowdhury and Raouf Boutaba. A survey of network virtualization. *Computer Networks*, 54(5):862–876, April 2010.

[Cha09] Chandola, Varun and Banerjee, Arindam and Kumar, Vipin. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, July 2009.

[Cl1] Ceilometer measurements. `https://docs.openstack.org/ceilometer/pike/admin/telemetry-measurements.html`.

[CL95] Yin-Wong Cheung and Kon S Lai. Lag order and critical values of the augmented dickey–fuller test. *Journal of Business & Economic Statistics*, 13(3):277–280, 1995.

[Coh09] Cohen, R. Cloud attack: Economic denial of sustainability (EDoS). `http://www.elasticvapor.com/2009/01/cloud-attack-economic-denial-of.html`, January 2009.

[CrF] CROWD Project: Connectivity Management for EneRgy Optimised Wireless Dense Networks", Funded under: FP7-ICT. Project Reference: 318115. `https://www.fp7-unify.eu/`.

[CRSG+15] Jose Costa-Requena, Jesús Llorente Santos, Vicent Ferrer Guasch, Kimmo Ahokas, Gopika Premsankar, Sakari Luukkainen, Oscar López Pérez, Mikel Uriarte Itzazelaia, Ijaz Ahmad, Madhusanka Liyanage, et al. SDN and NFV integration in generalized mobile network architecture. In *Proceedings of the European Conference on Networks and Communications (EuCNC)*, pages 154–158, Paris, France, June 2015.

[CSD15] Maria Mikela Chatzimichailidou, Neville A Stanton, and Ioannis M Dokas. The concept of risk situation awareness provision: towards a new approach for assessing the DSA about the threats and vulnerabilities of complex socio-technical systems. *Safety science*, 79:126–138, November 2015.

[CSJN05] J. Cao, D. P. Spooner, Stephen A. Jarvis, and Graham R. Nudd. Grid load balancing using intelligent agents. *Future generation computer systems*, 21(1):135–149, January 2005.

[CV17] Ángel Leonardo Valdivieso Caraguay and Luis Javier García Villalba. Monitoring and discovery for self-organized network management in virtualized and software defined networks. *Sensors*, 17(4):731, March 2017.

[Cv9] CVE-2016-9877. `https://www.cvedetails.com/cve/CVE-2016-9877/`.

[DC2] DICE: Developing Data-Intensive Cloud Applications with Iterative Quality Enhancement, Funded under: ICT-09-2014. Project reference: 644869.

[Den14a] D. E. Denning. Framework and principles for active cyber defense. *Computers & Security*, 40:108–113, February 2014.

[Den14b] Dorothy E Denning. Framework and principles for active cyber defense. *Computers & Security*, 40:108–113, February 2014.

[DLNW13] Hoang T Dinh, Chonho Lee, Dusit Niyato, and Ping Wang. A survey of mobile cloud computing: architecture, applications, and approaches. *Wireless communications and mobile computing*, 13(18):1587–1611, December 2013.

[DMR16] M. Díaz, C. Martin, and B. Rubio. State-of-the-art, challenges, and open issues in the integration of Internet of things and cloud computing. *Journal of Network and Computer Applications*, 67:99–117, May 2016.

[Dot15] Philip Doty. US homeland security and risk assessment. *Government Information Quarterly*, 32(3):342–352, July 2015.

[DRAP15] Gregory Ditzler, Manuel Roveri, Cesare Alippi, and Robi Polikar. Learning in nonstationary environments: A survey. *IEEE Computational Intelligence Magazine*, 10(4):12–25, November 2015.

[DSC] The CAIDA UCSD Anonymized Internet Traces 2016. `http://www.caida.org/data/passive/passive_2016_dataset.xml`.

[ECH] Horizon 2020: The EU Framework Programme for Research and Innovation. `https://ec.europa.eu/programmes/horizon2020/`.

[EKS+96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)*, volume 96, pages 226–231, Portland, OR, US, August 1996.

[End88] Mica R. Endsley. Design and evaluation for situation awareness enhancement. In *Proceedings of the Human Factors Society annual meeting*, volume 32, pages 97–101, Santa Monica, CA, US, October 1988.

[EP11] Ryan Elwell and Robi Polikar. Incremental learning of concept drift in nonstationary environments. *IEEE Transactions on Neural Networks*, 22(10):1517–1531, October 2011.

[ESE14] Saber M Elsayed, Ruhul A Sarker, and Daryl L Essam. A new genetic algorithm for solving optimization problems. *Engineering Applications of Artificial Intelligence*, 27:57–69, January 2014.

[EtM] Mobile Edge Computing A key technology towards 5G. `http://www.etsi.org/images/files/ETSIWhiteThesiss/etsi_wp11_mec_a_key_technology_towards_5g.pdf`.

[ETS13] ETSI. Network Functions Virtualization: Architectural Framework. `http://www.etsi.org/deliver/etsi_gs/nfv/001_099/002/01.01.01_60/gs_nfv002v010101p.pdf`, October 2013.

[Eur09] European Comission. Future Internet 2020: Visions of an Industry Expert Group. `http://ec.europa.eu/information_society/activities/foi/library/EPR.pdf`, May 2009.

[Eur14] European Technology Platform for Communications Networks and Services (NetWorld2020 ETP). Expert Working Group on 5G: Challenges, Research Priorities, and Recommendations. `https://networld2020.eu/wp-content/uploads/2014/02/NetWorld2020_Joint-Whitepaper-V8_public-consultation.pdf`, August 2014.

[FB14] Ulrik Franke and Joel Brynielsson. Cyber situational awareness–a systematic review of the literature. *Computers & Security*, 46:18–31, October 2014.

[FI92] Usama M. Fayyad and Keki B. Irani. On the handling of continuous-valued attributes in decision tree generation. *Machine learning*, 8(1):87–102, january 1992.

[FL14] Hongyinping Feng and Shengjia Li. Active disturbance rejection control based on weighed-moving-average-state-observer. *Journal of Mathematical Analysis and Applications*, 411(1):354–361, March 2014.

[FM09] C. Fortuna and M. Mohorcic. Trends in the development of communication networks: Cognitive networks. *Computer networks*, 53(9):1354–1376, June 2009.

[For82] Charles L Forgy. Rete: A fast algorithm for the many pattern/many object pattern match problem. *Artificial Intelligence*, 19(1):17–37, September 1982.

[FR15] Massimo Ficco and Massimiliano Rak. Stealthy denial of service strategy in cloud computing. *IEEE Transactions on Cloud Computing*, 3(1):80–94, March 2015.

[FRZ14] N. Feamster, J. Rexford, and E. Zegura. The Road to SDN: An Intellectual History of Programmable Networks. *ACM SIGCOMM Computer Communication Review*, 44(2):87–98, December 2014.

[GA13] Shahabeddin Geravand and Mahmood Ahmadi. Bloom filter applications in network security: A state-of-the-art survey. *Computer Networks*, 57(18):4047–4064, September 2013.

[Gar17] Gartner. Leading the IoT. `https://www.gartner.com/imagesrv/books/iot/iotEbook_digital.pdf`, 2017.

[GD91] David E Goldberg and Kalyanmoy Deb. A comparative analysis of selection schemes used in genetic algorithms. *Foundations of genetic algorithms*, 1:69–93, 1991.

[GJ06] Everette S Gardner Jr. Exponential smoothing: The state of the art—part ii. *International journal of forecasting*, 22(4):637–666, December 2006.

[GJD80] Everette S. Gardner Jr and David G. Dannenbring. Forecasting with exponential smoothing: Some guidelines for model selection. *Decision Sciences*, 11(2):370–383, April 1980.

[GP01] Thomer M Gil and Massimiliano Poletto. Multops: A data-structure for bandwidth attack detection. In *Proceedings of the 10th USENIX Security Symposium*, pages 23–38, Washington, DC, US, August 2001.

[GRA16] L. Gavrilovska, V. Rakovic, and V. Atanasovski. Visions Towards 5G: Technical Requirements and Potential Enablers. *Wireless Personal Communications*, 87(3):731–757, April 2016.

[Gro73] G. K. Groff. Empirical comparison of models for short range forecasting. *Management Science*, 20(1):22–31, September 1973.

[GVSOMV17] Luis Javier García Villalba, Ana Lucila Sandoval Orozco, and Jorge Maestre Vidal. Advanced payload analyzer preprocessor. *Future Generation Computer Systems*, 76:474–485, November 2017.

[HA14] Sven Ove Hansson and Terje Aven. Is risk analysis scientific? *Risk Analysis*, 34(7):1173–1183, June 2014.

[Hal99] Mark Andrew Hall. *Correlation-based feature selection for machine learning.* PhD thesis, April 1999.

[HBK15] Nazrul Hoque, Dhruba K Bhattacharyya, and Jugal K. Kalita. Botnet in ddos attacks: Trends and challenges. *IEEE Communications Surveys and Tutorials*, 17(4):2242–2270, First quarter 2015.

[HCT+14] Rick Hofstede, Pavel Celeda, Brian Trammell, Idilio Drago, Ramin Sadre, Anna Sperotto, and Aiko Pras. Flow monitoring explained: From packet capture to data analysis with netflow and ipfix. *IEEE Communications Surveys & Tutorials*, 16(4):2037–2064, May 2014.

[HFW08] Kathryn Hempstalk, Eibe Frank, and Ian H. Witten. One-class classification by combining density and class probability estimation. In *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 505–519, Antwerp, Belgium, September 2008.

[HGLFMV+18] A. Herranz González, B. Lorenzo Fernández, D. Maestre Vidal, G. Rius García, M. A. Sotelo Monge, J. Maestre Vidal, and L. J. García Villalba. DroidSentinel: ¿Está mi dispositivo móvil participando en un ataque DDoS? In *Proceedings of the Jornadas Nacionales de Investigación en Ciberseguridad (JNIC)*, pages 73–80, San Sebastian, Spain, June 2018.

[HH15] Ekram Hossain and Monowar Hasan. 5g cellular: key enabling technologies and research challenges. *IEEE Instrumentation & Measurement Magazine*, 18(3):11–21, May 2015.

[HKOS05] R. J. Hyndman, A. B. Koehler, J. K. Ord, and R. D. Snyder. Prediction intervals for exponential smoothing state space models. Technical report, January 2005.

[HLSR18] Bin Hu, Xiaowei Li, Shuting Sun, and Martyn Ratcliffe. Attention recognition in eeg-based affective learning research using cfs+ knn algorithm. *IEEE/ACM transactions on computational biology and bioinformatics*, 15(1):38–45, January 2018.

[Hof08] C. Hoff. Cloud Computing Security: From DDoS (Distributed Denial Of Service) to EDoS (Economic Denial of Sustainability). `http://rationalsecurity.typepad.com/blog/2008/11/cloud-computing-security-from-ddos-distributed-denial-of-service-to-edos-economic-denial-of-sustaina.html`, November 2008.

[Hof09] C. Hoff. A Couple of Follow-Ups On The EDoS (Economic Denial Of Sustainability) Concept... `http://rationalsecurity.typepad.com/blog/edos/`, January 2009.

[Hol93] Robert C Holte. Very simple classification rules perform well on most commonly used datasets. *Machine learning*, 11(1):63–90, April 1993.

[Hol04] C. C. Holt. Forecasting seasonals and trends by exponentially weighted moving averages. *International journal of forecasting*, 20(1):5–10, March 2004.

[HPSRFRAB09] Elena Hernández-Pereira, Juan A Suárez-Romero, Oscar Fontenla-Romero, and Amparo Alonso-Betanzos. Conversion methods for symbolic features: A comparison applied to an intrusion detection problem. *Expert Systems with Applications*, 36(7):10612–10617, September 2009.

[HSMA14] Hassan Hawilo, Abdallah Shami, Maysam Mirahmadi, and Rasool Asal. NFV: state of the art, challenges, and implementation in next generation mobile networks (vEPC). *IEEE Network*, 28(6):18–26, November 2014.

[HSS12a] S. Hämäläinen, H. Sanneck, and C. Sartori. *LTE self-organising networks (SON): network management automation for operational efficiency*. John Wiley & Sons, January 2012.

[HSS12b] S. Hamalainen, S. Sanneck, and C. Sartori. *LTE self-organising networks (SON): network management automation for operational efficiency*. John Wiley & Sons, Hoboken, NJ, US, 2012.

[HT82] Steven C Hillmer and George C Tiao. An arima-model-based approach to seasonal adjustment. *Journal of the American Statistical Association*, 77(377):63–70, October 1982.

[HVV17] Pilar Holgado, VICTOR A VILLAGRA, and Luis Vazquez. Real-time multistep attack prediction based on hidden markov models. *IEEE Transactions on Dependable and Secure Computing*, September 2017.

[Int13] International Organization for Standardization. ISO/IEC 27002: Information technology – Security techniques – Code of practice for information security management. `http://www.iso.org/iso/catalogue_detail?csnumber=54533`, October 2013.

[IT11] Joseph Idziorek and Mark Tannian. Exploiting cloud utility models for profit and ruin. In *Proceedings of the IEEE International Conference on Cloud Computing (CLOUD)*, pages 33–40, Washington, DC, USA, July 2011.

[ITJ12] Joseph Idziorek, Mark Tannian, and Doug Jacobson. Attribution of fraudulent resource consumption in the cloud. In *Proceedings of the IEEE 5th International Conference on Cloud Computing (CLOUD)*, pages 99–106, Honolulu, HI, US, June 2012.

[JBMS99] FR Johnston, JE Boyland, M Meadows, and E Shale. Some properties of a simple moving average when applied to forecasting a time series. *Journal of the Operational Research Society*, 50(12):1267–1271, December 1999.

[JCG+17] Mohammad Hanif Jhaveri, Orcun Cetin, Carlos Gañán, Tyler Moore, and Michel Van Eeten. Abuse reporting and the fight against cybercrime. *ACM Computing Surveys (CSUR)*, 49(4):68, February 2017.

[JL95] George H John and Pat Langley. Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 338–345, Montreal, Canada, August 1995.

[JL14] EunHee Jeong and ByungKwan Lee. An ip traceback protocol using a compressed hash table, a sinkhole router and data mining based on network forensics against network attacks. *Future Generation Computer Systems*, 33:42–52, April 2014.

[JST+07] Zhou Jian, Haruhiko Shirai, Isamu Takahashi, Jousuke Kuroiwa, Tomohiro Odaka, and Hisakazu Ogura. Masquerade detection by boosting decision stumps using unix commands. *computers & security*, 26(4):311–318, June 2007.

[JSTD16] Khundrakpam Johnson Singh, Khelchandra Thongam, and Tanmay De. Entropy-based application layer ddos attack detection using artificial neural networks. *Entropy*, 18(10):350, October 2016.

[JVDMB+07] B. Jennings, S. Van Der Meer, S. Balasubramaniam, D. Botvich, M. Foghlú, W. Donnelly, and J. Strassner. Towards autonomic management of communications networks. *IEEE Communications Magazine*, 45(10):112–121, October 2007.

[JW13] Ruofan Jin and Bing Wang. Malware detection for mobile devices using software-defined networking. In *Proceedings of the Second GENI Research and Educational Experiment Workshop*, pages 81–88, Salt Lake City, UT, US, March 2013.

[K+95] Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the International Joint Conference on AI*, volume 14, pages 1137–1145, Montreal, Quebec, Canada, August 1995.

[KBP14] Ankunda R Kiremire, Matthias R Brust, and Vir V Phoha. Using network motifs to investigate the influence of network topology on ppm-based ip traceback schemes. *Computer Networks*, 72:14–32, October 2014.

[Kea16] J. Kwon and et. al. Psybog: A scalable botnet detection method for large-scale dns traffic. *Computer Networks*, 97:48–73, March 2016.

[KN09] Soon Hin Khor and Akihiro Nakao. spow: On-demand cloud-based eddos mitigation mechanism. In *Proceedings of the IEEE/IFIP International Conference on Dependable Systems & Networks (DSN)*, pages 1–6, Lisbon, Portugal, June 2009.

[KNB13] Anusha Koduru, TulasiRam Neelakantam, and S Mary Saira Bhanu. Detection of economic denial of sustainability using time spent on a web page in cloud. In *Proceedings of the IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, pages 1–4, Bangalore, India, October 2013.

[KRG01] Ali Kamrani, Wang Rong, and Ricardo Gonzalez. A genetic algorithm methodology for data mining and intelligent knowledge acquisition. *Computers & Industrial Engineering*, 40(4):361–377, September 2001.

[KSK+12] Madarapu Naresh Kumar, P Sujatha, Vamshi Kalva, Rohit Nagori, Anil Kumar Katukojwala, and Mukesh Kumar. Mitigating economic denial of sustainability (edos) in cloud computing using in-cloud scrubber service. In *Proceedings of the Fourth International Conference on Computational Intelligence and Communication Networks (CICN)*, pages 535–539, Mathura, India, November 2012.

[KVF+12] Sanjeev Khanna, Santosh S Venkatesh, Omid Fatemieh, Fariba Khan, and Carl A Gunter. Adaptive selective verification: An efficient adaptive countermeasure to thwart dos attacks. *IEEE/ACM Transactions on Networking (TON)*, 20(3):715–728, October 2012.

[KW08] Michael BC Khoo and VH Wong. A double moving average control chart. *Communications in Statistics—Simulation and Computation®*, 37(8):1696–1708, October 2008.

[LDSR05] Pavel Laskov, Patrick Düssel, Christin Schäfer, and Konrad Rieck. Learning intrusion detection: supervised or unsupervised? In *Proceedings of the International Conference on Image Analysis and Processing*, pages 50–57, Cagliari, Italy, September 2005.

[Lee17] Lee, J.K. and Moon, S.Y. and Park, J.H. CloudRPS: a cloud analysis based enhanced ransomware prevention system. *The Journal of Supercomputing*, 73:3065–6084, July 2017.

[LII+15] Hafiz Yasar Lateef, Ali Imran, Muhammad Ali Imran, Lorenza Giupponi, and Mischa Dohler. Lte-advanced self-organizing network conflicts and coordination algorithms. *IEEE Wireless Communications*, 22(3):108–117, June 2015.

[LLZZ13] Hongbin Luo, Yi Lin, Hongke Zhang, and Moshe Zukerman. Preventing ddos attacks by identifier/locator separation. *IEEE network*, 27(6):60–65, December 2013.

[LM15] Yu-Beng Leau and Selvakumar Manickam. Network security situation prediction: a review and discussion. In *Proceedings of the International Conference on Soft Computing, Intelligence Systems, and Information Technology*, pages 424–435, Bali, Indonesia, March 2015.

[LMKY16] Yuliang Li, Rui Miao, Changhoon Kim, and Minlan Yu. Flowradar: A better netflow for data centers. In *Proceedings of the 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI6)*, pages 311–324, Santa Clara, CA, March 2016.

[LVV17] Lorena Isabel Barona López, Jorge Maestre Vidal, and Luis Javier García Villalba. Orchestration of use-case driven analytics in 5G scenarios. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–21, July 2017.

[MAB+08] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. OpenFlow: Enabling Innovation in Campus Networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74, March 2008.

[MAJ13] Seyed Ali Mirheidari, Sajjad Arshad, and Rasool Jalili. Alert correlation algorithms: A survey and taxonomy. In *Proceedings of the 5th International Symposium on Cyberspace Safety and Security (CSS)*, volume 8300, pages 183–197, Zhangjiajie, China, November 2013.

[MAJ14] Lakshay Malhotra, Devyani Agarwal, and Arunima Jaiswal. Virtualization in cloud computing. *J. Inform. Tech. Softw. Eng*, 4(2), June 2014.

[Mak00] Makridakis, Spyros and Hibon, Michele. The M3-Competition: results, conclusions and implications. *International journal of forecasting*, 16(4):451–476, December 2000.

[MARH13] Muddassar Masood, Zahid Anwar, Syed Ali Raza, and Muhammad Ali Hur. Edos armor: a cost effective economic denial of sustainability attack mitigation framework for e-commerce applications in cloud environments. In *Proceedings of the 16th International Multi Topic Conference (INMIC)*, pages 37–42, Lahore, Pakistan, December 2013.

[Mas09] Masao Masugi. Applying a recurrence plot scheme to analyze non-stationary transition patterns of ip-network traffic. *Communications in Nonlinear Science and Numerical Simulation*, 14(4):1418–1430, April 2009.

[McC91] J. McCumber. Information Systems Security: A Comprehensive Model. In *Proceedings of the 14th NIST National Computer Security Conference*, pages 328–337, Washington, D.C. USA, October 1991.

[McH00] John McHugh. Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory. *ACM Transactions on Information and System Security (TISSEC)*, 3(4):262–294, November 2000.

[MDB17] Oleg Markelov, Viet Nguyen Duc, and Mikhail Bogachev. Statistical modeling of the Internet traffic dynamics: To which extent do we need long-term correlations? *Physica A: Statistical Mechanics and its Applications*, 485:48–60, November 2017.

[MDML17] Vincenzo Matta, Mario Di Mauro, and Maurizio Longo. Ddos attacks with randomized traffic innovation: botnet identification challenges and strategies. *IEEE Transactions on Information Forensics and Security*, 12(8):1844–1859, August 2017.

[Men15] Paulo Mendes. Combining data naming and context awareness for pervasive networks. *Journal of Network and Computer Applications*, 50:114–125, April 2015.

[MGC+18] Lorenzo Fernández Maimó, Ángel Luis Perales Gómez, Félix J García Clemente, Manuel Gil Pérez, and Gregorio Martínez Pérez. A self-adaptive deep learning-based system for anomaly detection in 5g networks. *IEEE Access*, 6:7700–7712, February 2018.

[MKK11] Syed Akbar Mehdi, Junaid Khalid, and Syed Ali Khayam. Revisiting traffic anomaly detection using software defined networking. In *Proceedings of the 14th International Symposium on Recent Advances in Intrusion Detection (RAID)*, pages 161–180, Menlo Park, CA, US, September 2011.

[MMA] MAGERIT: Risk Analysis and Management Methodology for Information Systems. `http://administracionelectronica.gob.es/pae_Home/pae_Documentacion/pae_Metodolog/pae_Magerit.html?idioma=en`.

[Moo96] Todd K Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60, November 1996.

[MPR03] Jelena Mirkovic, Gregory Prier, and Peter Reiher. Source-end ddos defense. In *Proceeding of Second IEEE International Symposium on the Network Computing and Applications*, pages 171–178, Cambridge, MA, US, April 2003.

[MSK17] Douglas C MacFarland, Craig A Shue, and Andrew J Kalafut. The best bang for the byte: Characterizing the potential of dns amplification attacks. *Computer Networks*, 116:12–21, April 2017.

[Mul94a] Patrick G. Mulloy. Smoothing data with faster moving averages. *Stocks & Commodities*, 12(1):11–19, January 1994.

[Mul94b] Patrick G. Mulloy. Smoothing data with less lag. *Technical Analysis of Stocks & Commodities*, 12:72–80, June 1994.

[MVSMGV18] J. Maestre Vidal, M. A. Sotelo Monge, and L. J. García Villalba. Detecting Workload-based and Instantiation-based Economic Denial of Sustainability on 5G environments. In *Proceedings of the 13th International Conference on Availability, Reliability and Security (ARES)*, pages 50:1–50:8, Hamburg, Germany, August 2018.

[MVSOGV18] Jorge Maestre Vidal, Ana Lucila Sandoval Orozco, and Luis Javier García Villalba. Adaptive artificial immune networks for mitigating dos flooding attacks. *Swarm and Evolutionary Computation*, 38:94–108, February 2018.

[MVTG14] J. Medved, R. Varga, A. Tkacik, and K. Gray. Opendaylight: Towards a Model-driven SDN Controller Architecture. In *Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, pages 1–6, Sydney, NSW, Australia, June 2014.

[MWH97] Spyros Makridakis, Steven C. Wheelwright, and Rob J. Hyndman. *Forecasting methods and applications*. John wiley & sons, December 1997.

[NA08] Thuy TT Nguyen and Grenville Armitage. A survey of techniques for internet traffic classification using machine learning. *IEEE Communications Surveys & Tutorials*, 10(4):56–76, Fourth quarter 2008.

[NCC+16] P. Neves, R. Calé, M. R. Costa, C. Parada, B. Parreira, J. Alcaraz-Calero, Q. Wang, J. Nightingale, E. Chirivella-Perez, and W. Jiang. The SELFNET Approach for Autonomic Management in an NFV/SDN Networking Paradigm. *International Journal of Distributed Sensor Networks*, 12(2):1–17, February 2016.

[NG13] T. D. Nadeau and K. Gray. *SDN: Software Defined Networks*. O'Reilly Media, August 2013.

[NGM15] NGMN Alliance. 5G White Paper. https://www.ngmn.org/fileadmin/ngmn/content/images/news/ngmn_news/NGMN_5G_White_Paper_V1_0.pdf, February 2015.

[NIS11] NIST. The NIST Definition of Cloud Computing. https://csrc.nist.gov/publications/detail/sp/800-145/final, September 2011.

[Nom08] Nomor Research. Self-organizing networks (SON) in 3GPP long term evolution. https://pdfs.semanticscholar.org/242a/5ec3915d3514969d7faab9dc23612745fe06.pdf, May 2008.

[NSP] NIST-SP800 series special publications on computer security. https://csrc.nist.gov/publications/sp800.

[NWAC+16] J. Nightingale, Q. Wang, J. M. Alcaraz Calero, E. Chirivella-Perez, and M. Ulbricht. QoE-driven, Energy-aware Video Adaptation in 5G Networks: The SELFNET Self-optimisation Use Case. *International Journal of Distributed Sensor Networks*, 1(12):1–15, January 2016.

[NZ1] Noiszy. https://noiszy.com.

[ÖB15] İlker Özçelik and Richard R Brooks. Deceiving entropy based dos detection. *Computers & Security*, 48:234–245, February 2015.

[OBB+14] Afif Osseiran, Federico Boccardi, Volker Braun, Katsutoshi Kusume, Patrick Marsch, Michal Maternia, Olav Queseth, Malte Schellmann, Hans Schotten, Hidekazu Taoka, et al. Scenarios for 5G mobile and wireless communications: the vision of the METIS project. *IEEE Communications Magazine*, 52(5):26–35, May 2014.

[ODT] OpenDaylight TSDR. https://wiki.opendaylight.org/view/Project_Proposals:Time_Series_Data_Repository.

[OFv] OpenFlow Switch Specification version 1.0.0. Technical report.

[OGIR14] Colin O'Reilly, Alexander Gluhak, Muhammad Ali Imran, and Sutharshan Rajasegarar. Anomaly detection in wireless sensor networks in a non-stationary environment. *IEEE Communications Surveys & Tutorials*, 16(3):1413–1432, Third quarter 2014.

[OHT05] Chorng-Shyong Ong, Jih-Jeng Huang, and Gwo-Hshiung Tzeng. Model identification of arima family using genetic algorithms. *Applied Mathematics and Computation*, 164(3):885–912, May 2005.

[ONF] ONF. Open Networking Foundation. https://www.opennetworking.org/.

[ONS12] Software-Defined Networking: The New Norm for Networks. White Paper, Open Networking Foundation, April 2012.

[Ope] Openstack. Open source sotware for creating private and public clouds. https://www.openstack.org.

[OSL] Openstack Senlin: Clustering service for OpenStack clouds. https://wiki.openstack.org/wiki/Senlin.

[P5C] CogNet: Building an Intelligent System of Insights and Action for 5G Network Management. http://www.cognet.5g-ppp.eu.

[P5S] Self-Organized Network Management in Virtualized and Software Defined Networks (SELFNET). Project reference: H2020-ICT- 2014-2/671672, Funded under: H2020. http://www.selfnet-5g.eu.

[PCCT14] Marco AF Pimentel, David A Clifton, Lei Clifton, and Lionel Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, June 2014.

[PCM] CHARISMA Project: Converged Heterogeneous Advanced 5G Cloud-RAN Architecture for Intelligent and Secure Media Access. Funded under: H2020-ICT-2014-2. Project Reference: 671704. http://www.charisma5g.eu.

[PDG+16] Maria Rita Palattella, Mischa Dohler, Alfredo Grieco, Gianluca Rizzo, Johan Torsner, Thomas Engel, and Latif Ladid. Internet of things in the 5g era: Enablers, architecture, and business models. *IEEE Journal on Selected Areas in Communications*, 34(3):510–527, February 2016.

[PEq] Equinix, Inc. http://www.equinix.com.

[Pir14] Pekka Pirinen. A brief overview of 5G research activities. In *Proceedings of the 1st International Conference on 5G for Ubiquitous Connectivity (5GU)*, pages 17–22, Akaslompolo, Finland, November 2014.

[PLW+13] M. Peng, D. Liang, Y. Wei, J. Li, and H. Chen. Self-configuration and self-optimization in LTE-advanced heterogeneous networks. *IEEE Communications Magazine*, 51(5):36–45, May 2013.

[POv] Open vSwitch. https://www.openvswitch.org.

[PP98] Donn B Parker and Donn B Parker. *Fighting computer crime: A new framework for protecting information.* John Wiley & Sons, September 1998.

[PTn] T-NOVA Project: Network Functions as-a-Service over Virtualised Infrastructures. http://www.t-nova.eu.

[PZCG14] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos. Context aware computing for the internet of things: A survey. *IEEE communications surveys and tutorials*, 16(1):414–454, January 2014.

[R5A] 5G Americas. http://www.5gamericas.org.

[R5G] 5G-Ensure Project (2014) Enablers for Network and System Security and Resilience. Funded under: H2020-ICT-2014-2. Project Reference: 671562. http://www.5gensure.eu.

[RbM] RabbitMQ: Messaging that just works. https://www.rabbitmq.com.

[ReC] CVSS: Common Vulnerability Scoring System. https://www.first.org/cvss/specification-document.

[RJPD14a] Leszek Rutkowski, Maciej Jaworski, Lena Pietruczuk, and Piotr Duda. The cart decision tree for mining data streams. *Information Sciences*, 266:1–15, May 2014.

[RJPD14b] Leszek Rutkowski, Maciej Jaworski, Lena Pietruczuk, and Piotr Duda. The cart decision tree for mining data streams. *Information Sciences*, 266:1–15, May 2014.

[RK15] Ori Rottenstreich and Isaac Keslassy. The bloom paradox: When not to use a bloom filter. *IEEE/ACM Transactions on Networking (TON)*, 23(3):703–716, June 2015.

[RL06] Konrad Rieck and Pavel Laskov. Detecting unknown network attacks using language models. In *Proceedings of the 3rd International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)*, pages 74–90, Berlin, Germany, July 2006.

[Rok10] Lior Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1):1–39, February 2010.

[ROM01] Gunnar Ratsch, Takashi Onoda, and K-R Muller. Soft margins for AdaBoost. *Machine learning*, 42(3):287–320, March 2001.

[RPI] IMT-2020 5G Promotion Group. `http://www.imt-2020.cn/en/introduction`.

[RPM] METIS-II: Mobile and wireless communications Enablers for Twenty-twenty (2020) Information Society-II. Funded under: H2020-ICT-2014-2. `https://www.metis2020.com`.

[RPU] UNIFY Project: Unifying Cloud and Carrier Networks. Project reference: 619609. Funded under: FP7. `https://www.fp7-unify.eu/`.

[RS2] SONATA: Service Programming and Orchestration for Virtualized Software Networks. `http://www.sonata-nfv.eu`.

[Rui16] Ruiz, Germán Ramos and Bandera, Carlos Fernández and Temes, Tomás Gómez-Acebo and Gutierrez, Ana Sánchez-Ostiz. Genetic algorithm for building envelope calibration. *Applied Energy*, 168:691–705, April 2016.

[SC17] Ashish Singh and Kakali Chatterjee. Cloud security issues and challenges: A survey. *Journal of Network and Computer Applications*, 79:88–115, February 2017.

[SD84] Said E Said and David A Dickey. Testing for unit roots in autoregressive-moving average models of unknown order. *Biometrika*, 71(3):599–607, December 1984.

[SDB] Drools: Business Rules Management System (BRMS). `https://www.drools.org`.

[SF00] D Senie and P Ferguson. Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing. RFC, IETF Network Working Group, May 2000.

[SFl] Flask-A Python Microframework. `http://flask.pocoo.org`.

[SG99] Chris Stauffer and W. Eric L. Grimson. Adaptive background mixture models for real-time tracking. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, page 2246, Fort Collins, CO, US, June 1999.

[SGS+17] Gaurav Somani, Manoj Singh Gaur, Dheeraj Sanghi, Mauro Conti, and Rajkumar Buyya. Ddos attacks in cloud computing: Issues, taxonomy, and future directions. *Computer Communications*, 107:30–48, July 2017.

[SGSC16] Gaurav Somani, Manoj Singh Gaur, Dheeraj Sanghi, and Mauro Conti. DDoS attacks in cloud computing: collateral damage to non-targets. *Computer Networks*, 109:157–171, November 2016.

[Sha48] C. E. Shannon. A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–656, October 1948.

[SIN] Internet Noise. `http://makeinternetnoise.com`.

[SK09] Nancy Samaan and Ahmed Karmouch. Towards autonomic network management: an analysis of current and future research directions. *IEEE Communications Surveys & Tutorials*, 11(3), August 2009.

[SL91] S Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674, June 1991.

[SLo] Low Orbit Ion Cannon (LOIC). `https://sourceforge.net/projects/loic/files`.

[SMFDV13] Saeed Salah, Gabriel Maciá-Fernández, and JesúS E DíAz-Verdejo. A model-based survey of alert correlation techniques. *Computer Networks*, 57(5):1289–1317, April 2013.

[SMGV17] M. A. Sotelo Monge and L. J. García Villalba. Arquitectura para la mejora de la calidad de servicios multimedia en redes SDN. In *Proceedings of the XXXII Simposium Nacional de la Unión Científica Internacional de Radio (URSI)*, Cartagena, Spain, September 2017.

[Smi12] Richard E Smith. A contemporary look at Saltzer and Schroeder's 1975 design principles. *IEEE Security & Privacy*, 10(6):20–25, November 2012.

[SMMVGV17a] M. A. Sotelo Monge, J. Maestre Vidal, and L. J. García Villalba. Entropy-Based Economic Denial of Sustainability Detection. *Entropy*, 19(12):649–649, November 2017.

[SMMVGV17b] M. A. Sotelo Monge, J. Maestre Vidal, and L. J. García Villalba. Knowledge Inference Analysis Framework for Incidence Management in 5G Networks. In *Proceedings of the 2nd IEEE European Symposium on Security and Privacy (EuroS&P)*, Paris, Framce, April 2017.

[SMMVGV17c] M. A. Sotelo Monge, J. Maestre Vidal, and L. J. García Villalba. Marco para el Análisis e Inferencia de Conocimiento en Redes 5G. In *Proceedings of the XIII Jornadas de Ingeniería Telemática (JITEL)*, Valencia, Spain, September 2017.

[SMMVGV17d] M. A. Sotelo Monge, J. Maestre Vidal, and L. J. García Villalba. Reasoning and Knowledge Acquisition Framework for 5G Network Analytics. *Sensors*, 17(10):2405–2405, October 2017.

[SMMVGV18a] M. A. Sotelo Monge, J. Maestre Vidal, and L. J. García Villalba. A novel Self-Organizing Network solution towards Crypto-ransomware Mitigation. In *Proceedings of the 13th International Conference on Availability, Reliability and Security (ARES)*, pages 48:1–48:10, Hamburg, Germany, August 2018.

[SMMVGV18b] M. A. Sotelo Monge, J. Maestre Vidal, and L. J. García Villalba. Detección de ataques de Denegación de Sostenibilidad Económica en redes Autoorganizadas. In *Proceedings of the Jornadas Nacionales de Investigación en Ciberseguridad (JNIC)*, pages 21–28, San Sebastian, Spain, June 2018.

[SMR14] Parminder Singh, Selvakumar Manickam, and Shafiq Ul Rehman. A survey of mitigation techniques against economic denial of sustainability (edos) attack on cloud computing architecture. In *Proceedings of the 3rd International Conference on Reliability, Infocom Technologies and Optimization (ICRITO)*, pages 1–4, Noida, India, October 2014.

[SOR16] Alan Saied, Richard E Overill, and Tomasz Radzik. Detection of known and unknown ddos attacks using artificial neural networks. *Neurocomputing*, 172:385–393, January 2016.

[SRA+16] J. P. Santos, A. Rui, L. Andrade, A. L. Valdivieso Caraguay, L. I. Barona López, and M. A. Sotelo Monge. SELFNET Framework Self-healing Capabilities for 5G Mobile Networks. *Transactions on Emerging Telecommunications Technologies*, 27(9):1225–1232, September 2016.

[SSABC16] Alireza Shameli-Sendi, Rouzbeh Aghababaei-Barzegar, and Mohamed Cheriet. Taxonomy of information security risk assessment (ISRA). *Computers & security*, 57:14–30, March 2016.

[SSK17] Karanpreet Singh, Paramvir Singh, and Krishan Kumar. Application layer HTTP-GET flood DDoS attacks: Research landscape and challenges. *Computers & security*, 65:344–372, March 2017.

[SWXH15] Zhiyang Su, Ting Wang, Yu Xia, and Mounir Hamdi. Cemon: A cost-effective flow monitoring system in software defined networks. *Computer Networks*, 92:101–115, December 2015.

[SYY+13] Z. Sheng, S. Yang, Y. Yu, A. Vasilakos, J. Mccann, and K. Leung. A survey on the ietf protocol suite for the internet of things: standards, challenges, and opportunities. *IEEE Wireless Communications*, 20(6):91–98, December 2013.

[TBLG09] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. A detailed analysis of the KDD CUP 99 data set. In *Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, pages 1–6, Ottawa, ON, Canada, July 2009.

[TF1] TSFRESH: Time Series Feature extraction based on scalable hypothesis tests. `https://github.com/blue-yonder/tsfresh`.

[THt] The Httperf HTTP load generator. `https://github.com/httperf/httperf`.

[TKJ16] Tarik Taleb, Adlen Ksentini, and Riku Jantti. "anything as a service" for 5g mobile systems. *IEEE Network*, 30(6):84–91, October 2016.

[TnS] TrackMeNot: Resisting Surveillance in Web Search. `https://cs.nyu.edu/trackmenot`.

[TPL+16] H. Tullberg, P. Popovski, Z. Li, M. A. Uusitalo, A. Hoglund, O. Bulakci, M. Fallgren, and J. F. Monserrat del Río. The METIS 5G system concept: Meeting the 5G requirements. *IEEE Communications Magazine*, 54(12):132–139, December 2016.

[VS12] S VivinSandar and Sudhir Shenai. Economic denial of sustainability (edos) in cloud services using http and xml based ddos attacks. *International Journal of Computer Applications*, 41(20):11–16, March 2012.

[VZF17] Gernot Vormayr, Tanja Zseby, and Joachim Fabini. Botnet communication patterns. *IEEE Communications Surveys and Tutorials*, 19(4):2768–2796, Fist quarter 2017.

[WCC+17] Yating Wang, Ray Chen, Jin-Hee Cho, Ananthram Swami, and Kevin S Chan. Trust-based service composition and binding with multiple objective optimization in service-oriented mobile ad hoc networks. *IEEE Transactions on Services Computing*, 10(4):660–672, August 2017.

[WcD] WarChild DoS test suit. `https://github.com/Souhardya`.

[WCXJ13] Wei Wei, Feng Chen, Yingjie Xia, and Guang Jin. A rank correlation based detection against distributed reflection dos attacks. *IEEE Communications Letters*, 17(1):173–175, January 2013.

[WDMS17] Kun Wang, Miao Du, Sabita Maharjan, and Yanfei Sun. Strategic honeypot game model for distributed denial of service attacks in the smart grid. *IEEE Transactions on Smart Grid*, 8(5):2474–2482, September 2017.

[Win60] Peter R Winters. Forecasting sales by exponentially weighted moving averages. *Management science*, 6(3):324–342, April 1960.

[WIY03] Dwi H Widyantoro, Thomas R Ioerger, and John Yen. Tracking changes in user interests with a few relevance judgments. In *Proceedings of the 12th international conference on Information and knowledge management*, pages 548–551, New Orleans, LA, US, November 2003.

[WkB] Bayesian Networks in WEKA. `http://www.cs.waikato.ac.nz/~remco/weka_bn`.

[Wkl] WEKA. `http://www.cs.waikato.ac.nz/ml/weka`.

[WLJW16] Lei Wang, Qing Li, Yong Jiang, and Jianping Wu. Towards mitigating link flooding attack via incremental sdn deployment. In *Proceedings of the IEEE Symposium on Computers and Communication (ISCC)*, pages 397–402, Messina, Italy, June 2016.

[WMLW18] Chenxu Wang, Tony T. N. Miu, Xiapu Luo, and Jinhe Wang. Skyshield: A sketch-based defense system against application layer DDoS attacks. *IEEE Transactions on Information Forensics and Security*, 13(3):559–573, March 2018.

[WMs] Microsoft Azure. `https://azure.microsoft.com`.

[WOS] Telemetry service overview. `https://docs.openstack.org/mitaka/install-guide-ubuntu/common/get_started_telemetry.html`.

[XSZ+15] Ming Xia, Meral Shirazipour, Ying Zhang, Howard Green, and Attila Takacs. Network function placement for NFV chaining in packet/optical datacenters. *Journal of Lightwave Technology*, 33(8):1565–1570, January 2015.

[XWY+17] Le Xiao, Wei Wei, Weidong Yang, Yulong Shen, and Xianglin Wu. A protocol-free detection against cloud oriented reflection dos attacks. *Soft Computing*, 21(13):3713–3721, July 2017.

[YBV15] Guang Yao, Jun Bi, and Athanasios V Vasilakos. Passive ip traceback: Disclosing the locations of ip spoofers from path backscatter. *IEEE Transactions on Information Forensics and Security*, 10(3):471–484, March 2015.

[YHSH17] Xiangrui Yang, Biao Han, Zhigang Sun, and Jinfeng Huang. Sdn-based ddos attack detection with cross-plane collaboration and lightweight flow monitoring. In *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, Singapore, Singapore, December 2017.

[YTGW14] Shui Yu, Yonghong Tian, Song Guo, and Dapeng Oliver Wu. Can we beat ddos attacks in clouds? *IEEE Transactions on Parallel and Distributed Systems*, 25(9):2245–2254, September 2014.

[YYGL16] Qiao Yan, F Richard Yu, Qingxiang Gong, and Jianqiang Li. Software-defined networking (sdn) and distributed denial of service (ddos) attacks in cloud computing environments: A survey, some research issues, and challenges. *IEEE Communications Surveys & Tutorials*, 18(1):602–622, First Quarter 2016.

[YZJ+12] Shui Yu, Wanlei Zhou, Weijia Jia, Song Guo, Yong Xiang, and Feilong Tang. Discriminating ddos attacks from flash crowds using flow correlation coefficient. *IEEE Transactions on Parallel and Distributed Systems*, 23(6):1073–1080, June 2012.

[YZV16] Zheng Yan, Peng Zhang, and Athanasios V Vasilakos. A security and trust framework for virtualized networks and software-defined networking. *Security and communication networks*, 9(16):3059–3069, November 2016.

[Zah17] Zahra, A. and Shah, M.A. IoT based ransomware growth rate evaluation and detection using command and control blacklisting. In *Proceedings of the 23rd International Conference on Automation and Computing*, pages 1–6, Huddersfield, UK, October 2017.

[ZCB10] Qi Zhang, Lu Cheng, and Raouf Boutaba. Cloud computing: state-of-the-art and research challenges. *Journal of internet services and applications*, 1(1):7–18, May 2010.

[Zim14] Zimek, Arthur and Campello, Ricardo JGB and Sander, Jörg. Ensembles for unsupervised outlier detection: challenges and research questions a position paper. *Acm Sigkdd Explorations Newsletter*, 15(1):11–22, June 2014.

[ZJT13] Saman Taghavi Zargar, James Joshi, and David Tipper. A survey of defense mechanisms against distributed denial of service (ddos) flooding attacks. *IEEE communications surveys & tutorials*, 15(4):2046–2069, March 2013.

[ZJW+14] Wei Zhou, Weijia Jia, Sheng Wen, Yang Xiang, and Wanlei Zhou. Detection and defense of application-layer DDoS attacks in backbone web traffic. *Future Generation Computer Systems*, 38:36–46, September 2014.

[ZWHL16] Peng Zhang, Huanzhao Wang, Chengchen Hu, and Chuang Lin. On denial of service attacks in software defined networks. *IEEE Network*, 30(6):28–33, December 2016.

Appendix A

List of Papers

1. M. A. Sotelo Monge, J. Maestre Vidal, L. J. García Villalba, "Knowledge Inference Analysis Framework for Incidence Management in 5G Networks", In the 2nd IEEE European Symposium on Security and Privacy (EuroS&P), Paris, France, April 26 - 28, 2017.

2. M. A. Sotelo Monge, L. J. García Villalba, "Arquitectura para la mejora de la calidad de servicios multimedia en redes SDN", In the XXXII Simposium Nacional de la Unión Científica Internacional de Radio (URSI), Cartagena, Spain, September 6-8, 2017.

3. M. A. Sotelo Monge, J. Maestre Vidal, L. J. García Villalba, "Marco para el Análisis e Inferencia de Conocimiento en Redes 5G", In the XIII Jornadas de Ingeniería Telemática (JITEL), Valencia, Spain, September 27-29, 2017.

4. M. A. Sotelo Monge, J. Maestre Vidal, L. J. García Villalba, "Reasoning and Knowledge Acquisition Framework for 5G Network Analytics". Sensors, Vol. 17, No. 10, pp. 2405-2405, October 2017.

5. M. A. Sotelo Monge, J. Maestre Vidal, L. J. García Villalba, "Entropy-Based Economic Denial of Sustainability Detection". Entropy, Vol. 19, No. 12, pp. 649-649, November 2017.

6. M. A. Sotelo Monge, J. Maestre Vidal, L. J. García Villalba, "Detección de ataques de Denegación de Sostenibilidad Económica en redes Autoorganizadas", Jornadas Nacionales de Investigación en Ciberseguridad (JNIC 2018), San Sebastian, Spain, June 13-15, 2018.

7. A. Herranz González, B. Lorenzo Fernández, D. Maestre Vidal, G. Rius García, M. A. Sotelo Monge, J. Maestre Vidal, L. J. García Villalba, "DroidSentinel: ¿Está mi dispositivo móvil participando en un ataque DDoS?", Jornadas Nacionales de Investigación en Ciberseguridad (JNIC 2018), San Sebastian, Spain, June 13-15, 2018.

8. J. Maestre Vidal, M. A. Sotelo Monge, L. J. García Villalba, "Detecting Workload-based and Instantiation-based Economic Denial of Sustainability on 5G

environments", Proceedings of the 13th International Conference on Availability, Reliability and Security (ARES 2018), Hamburg, Germany, August 27-30, 2018.

9. M. A. Sotelo Monge, J. Maestre Vidal, L. J. García Villalba, "A novel Self-Organizing Network solution towards Crypto-ransomware Mitigation", Proceedings of the 13th International Conference on Availability, Reliability and Security (ARES 2018), Hamburg, Germany, August 27-30, 2018.

Additional Publications

1. J. P. Santos, A. Rui, L. Andrade, A. L. Valdivieso Caraguay, L. I. Barona López, L. J. García Villalba, "SELFNET Framework Self-healing Capabilities for 5G Mobile Networks". Transactions on Emerging Telecommunications Technologies, Vol. 27, No. 9, pp. 1225-1232, June 2016.

2. P. Bisson, J. Wary, M. A. Sotelo Monge, et. al. "5G PPP Phase 1 Security Landscape", 5G PPP Security Work Group. June 2017.